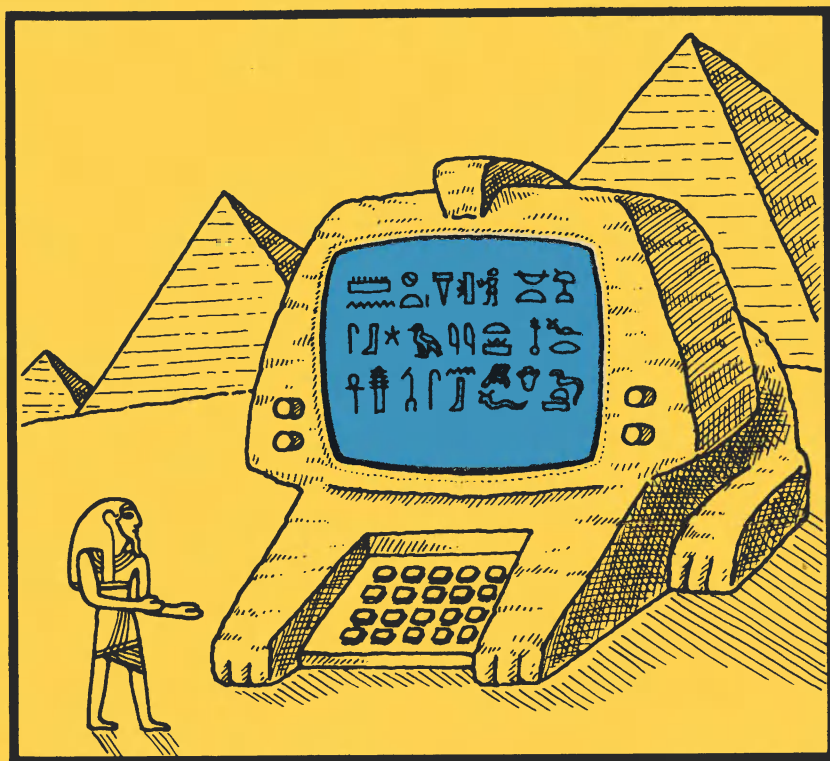


В.В.АЛЕКСАНДРОВ  
В.Н.АРСЕНТЬЕВ  
А.В.АРСЕНТЬЕВА

# Что может ЭВМ?



НАУЧНО-ПОПУЛЯРНАЯ БИБЛИОТЕКА ШКОЛЬНИКА

НАУЧНО-ПОПУЛЯРНАЯ БИБЛИОТЕКА ШКОЛЬНИКА

В. В. АЛЕКСАНДРОВ  
В. Н. АРСЕНТЬЕВ  
А. В. АРСЕНТЬЕВА

# Что может ЭВМ?

*Под общей редакцией  
д-ра техн. наук В. В. Александрова*



ЛЕНИНГРАД  
„МАШИНОСТРОЕНИЕ“  
ЛЕНИНГРАДСКОЕ ОТДЕЛЕНИЕ  
1987

ББК 32.973  
А46  
УДК 681.3

Рецензент канд. физ.-мат. наук В. В. Шапкин

**Александров В. В. и др.**

А46      Что может ЭВМ?/В. В. Александров, В. Н. Арсентьев, А. В. Арсентьева; Под общ. ред. д-ра техн. наук В. В. Александрова.— Л.: Машиностроение. Ленингр. отд-ние, 1988.— 124 с.: ил.— (Науч.-попул. б-ка школьника).

Что может и зачем нужна ЭВМ? Как возникла и развивалась идея создания «интеллектуального помощника» человека? Каковы социальная роль и последствия широкого внедрения ЭВМ? Этим вопросам посвящена книга. Одновременно с историей создания вычислительной машины рассказано об основных ее элементах, о языке взаимодействия человека с ЭВМ.

Книга рассчитана на учащихся старших классов и ПТУ.

А 240500000-940 174-86  
038(01)-88

ББК 32.973

## ПРЕДИСЛОВИЕ

«Посадка прошла как по маслу, несмотря на капризы гравитации, причиной которых были два солнца и шесть лун. Низкая облачность могла бы вызвать осложнения, если бы посадка была визуальной. Но Джексон (пилот космического корабля — В. А.) считал это ребячеством. Гораздо проще и безопасней было включить компьютер, откинуться в кресле и наслаждаться полетом.»  
(Из научно-фантастического рассказа Р. Шекли «Потолкуем малость»)

«Космонавты ... для окончательной стыковки со станцией перешли на ручное управление»; «...Впервые в отечественной практике осуществили стыковку транспортного корабля с неуправляемой станцией и восстановили ее работоспособность.»  
(Правда, 1985, 31 дек. )

Предвидения писателей-фантастов в целом оправдываются. Действительно, с ЭВМ связаны многие достижения современной науки и техники, в частности освоение космического пространства, автоматизация процессов умственного и физического труда, организация «безлюдного» производства.

Сегодня актуальными стали вопросы «что может ЭВМ?», «каковы ее роль и место в обществе?». Едва ли не ежедневно со страниц массовых периодических изданий на читателей обрушивается информация, которую, пользуясь принципом афористичности, можно изложить так: ЭВМ ставит диагноз, ЭВМ — мозг робота, ЭВМ играет в шахматы, ЭВМ проектирует и т. д. Авторы журнальных и газетных статей не скупятся на броские названия: «Компьютер о восстании на «Потемкине» (Знание — сила, 1985, № 12), «За прилавком — компьютер» (Правда, 1985, 31 дек.).

Общими усилиями средств массовой информации и научно-популярной литературы «создана» уникальная всемогущая суперЭВМ. Непосвященный читатель, слыша

отовсюду о невероятных возможностях компьютеров, невольно ассоциирует их избирательный образ с конкретной моделью. Но при первых же попытках ощутить самостоятельно всемогущество ЭВМ новичка ожидает разочарование.

Увы, ЭВМ не ставит диагноз, а только помогает врачу накапливать информацию и быстро проводить обследование по заранее выработанному критерию принятия решения. Да, ЭВМ определяет работоспособность робота и управляет им, но даже при полном функциональном совпадении движений робота и человека не следует делать вывод о тождественности принципов работы «мозга» ЭВМ и умственной деятельности человека. И программа ЭВМ, играющей в шахматы, не может быть адекватной мыслительному процессу шахматиста. Интересно, что понимание этого, казалось бы, очевидного факта пришло только после попыток при исследовании проблем создания искусственного интеллекта проимитировать творческие способности человека.

А в то же время популяризаторы зачастую оставляют в тени такие достоинства ЭВМ, как высокая скорость вычислений, огромная емкость памяти, быстрый поиск нужной информации. И это, по-видимому, объясняется ложной стыдливостью человека, нежеланием признать свою неспособность даже близко сравниться с ЭВМ в скорости счета, в длительном запоминании огромных объемов информации и др.

Итак, у ЭВМ есть и преимущества перед человеком, и недостатки, что порождает сложность ответа на вопрос «что может ЭВМ?». Специалисты в области вычислительной техники, как правило, подчеркивают только сильные стороны ЭВМ и считают, что для эффективного их использования надо знать программирование. Но программирование — только один из важнейших вопросов, касающихся работоспособности ЭВМ. Нельзя забывать о существовании таких проблем, как массовость и доступность ЭВМ, роль и место ее в обществе, действительный и мнимый прогресс в ее использовании, очередность и подготовленность внедрения ЭВМ в народное хозяйство, которые ждут своего разрешения.

На страницах этой книги мы рассмотрим проблему взаимодействия человека и ЭВМ, выявим важнейшие различия между так называемым естественным языком и языком, доступным машине. Попробуем проследить путь от этапа формулирования задачи человеком до получения решения на ЭВМ, чтобы выяснить, в какой мере этот этап

определяет эффективность всего процесса решения задачи на ЭВМ. Такие понятия, как «модель», «алгоритм», «программа», помогут оценить роль программиста и других специалистов, участвующих в процессе решения задач на ЭВМ.

Далее мы опишем общие для многих языков программирования элементы, к которым относятся прежде всего переменная, константа, функция, процедура, цикл, условие. Охарактеризуем наиболее употребимые языки программирования и средства диалога человека с машиной. А в заключение попытаемся определить граничные возможности ЭВМ на сегодняшнем этапе развития вычислительной техники, рассмотрим понятие «искусственный интеллект» и ответим на вопрос «чего не могут ЭВМ?».

Короче, мы предпримем попытку рассказать об ЭВМ с разных позиций и будем считать свою задачу выполненной, если наш читатель увидит в ЭВМ не «чудо», а удобное техническое средство, облегчающее деятельность человека, и при этом проникнется убеждением, что ЭВМ достойна стать одним из символов нынешнего столетия.

## ВВЕДЕНИЕ

Каменный век, бронзовый век, железный век..., промышленная революция. Каждое из этих емких понятий отражает яркий образ эпохи со свойственными ей экономическим укладом общества, материальным уровнем его развития, культурой и бытом.

Что же является характерным для нашего века — века автомобиля, электричества, авиации, атомной энергии, космонавтики, ЭВМ? Прежде всего — небывалые скорости развития науки, техники и технологии. Если период времени от использования средств, служивших для запоминания устных сообщений \*, до возникновения письменности поистине необозрим, то такие важные в истории человечества события, как изобретение книгопечатного станка (середина XV в.) и радиоприемника (1895 г.) разделены во времени примерно на 440 лет, создание радио и телевидения — на 30 лет. Сейчас разрыв между новым открытием и его практическим освоением чрезвычайно сократился. Так, внедрение транзистора заняло пять лет, а интегральных схем — три года.

Если еще в начале века присутствовали консерватизм и недоверие к техническим новшествам, то в наши дни, едва привыкнув к понятиям «робот», «манипулятор», мы говорим уже о «безлюдной» технологии, т. е. замене живого труда и относительно простых механизмов сложнейшими машинами.

Следует предостеречь читателя от узкого толкования слова «технология» как способа организации производст-

---

\* Например, древние индейские племена Северной Америки использовали для этих целей вавпумы — нити с нанизанными на них раковинами. Содержание передаваемых устно сообщений выражалось цветом, количеством и взаиморасположением раковин. — *Прим. ред.*

ва на фабриках и заводах. Конечно, прогресс в технологии производства облегчает труд и повышает его производительность. Но в рамках темы данной книги нас интересует, кроме того, технология, открывающая новые возможности в освоении природных ресурсов, технология научного поиска и эксперимента, технология проектирования и, естественно, проблема взаимодействия человека и ЭВМ.

Развитие технологии в широком смысле этого слова должно обеспечить уменьшение энергетических и материальных затрат при непременном увеличении объема продуктов труда. Постоянное расширение сферы действия новых технологий, обуславливающее необходимость в ускорении процессов накопления, обработки и передачи информации, принятия решений на ее основе, выступает объективной причиной небывало высокого уровня внедрения ЭВМ.

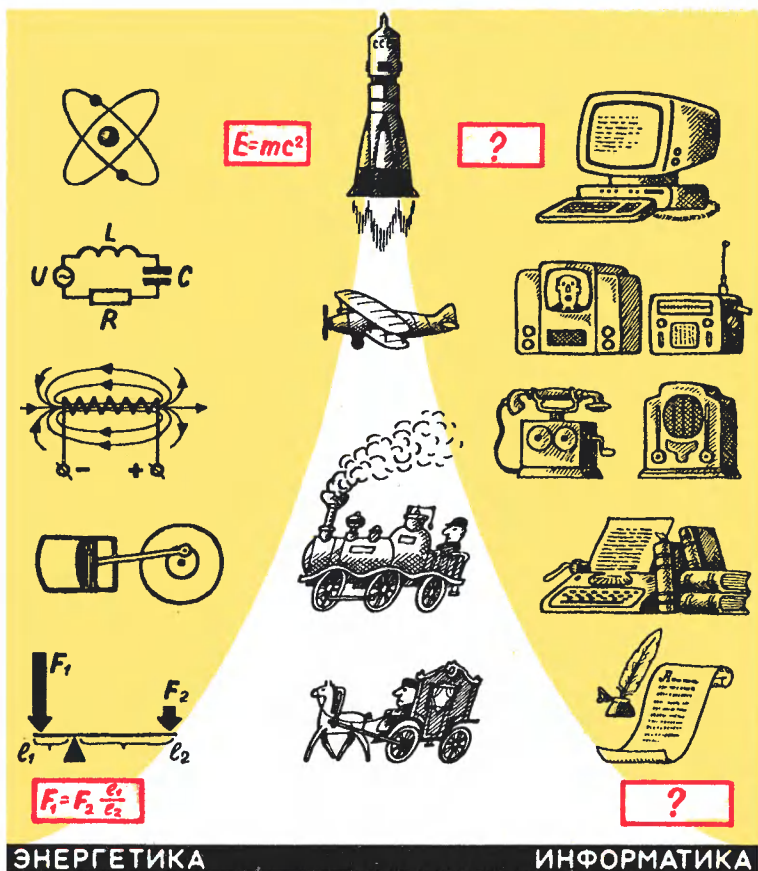
Пожалуй, легче перечислить те области науки и техники, где ЭВМ не применяется, а точнее, пока еще не применяется. ЭВМ используется в глубинах океана, не доступных человеку, и для управления космическими аппаратами. Вычислительная машина способна заменить человека в экстремальных ситуациях, угрожающих его здоровью и жизни. ЭВМ — это терпеливый помощник, охотно выполняющий нудные однотипные расчеты, не допуская при этом ошибок.

Стремительное развитие вычислительной техники объясняется тем, что сразу во многих областях деятельности человека возникла потребность в резком ускорении информационно-вычислительных процессов. ЭВМ позволяет принимать решения на основе огромного количества данных в кратчайший срок и с большой точностью. Ведь иногда бессмысленно производить точные вычисления, потому что время, затраченное на них, больше отводимого нам ситуацией. Например, управление крупным транспортным узлом, где чрезвычайно важно не выйти из графика и при этом объединить водный, автомобильный и железнодорожный транспорт в единую систему, сегодня уже невозможно без ЭВМ-диспетчера.

Существуют ЭВМ-дизайнеры, ЭВМ-архивариусы, ЭВМ-экзаменаторы, ЭВМ-водолазы. Этот перечень легко продолжить. Но, пожалуй, один из удивительнейших фактов заключается в том, что ЭВМ в значительной мере взяла на себя задачу создания себе подобных устройств, существенно повлияв на весь производственный процесс.

Быстрый прогресс наблюдается и в элементной базе вычислительной техники. Если для первых ЭВМ были ис-





*Сопоставление роста возможностей человека в области энергетики и в сфере информатики*

пользованы вакуумные и полупроводниковые элементы, то все возрастающие потребности в объеме вычислений и быстродействии потребовали принципиально иных решений. Ведь нельзя до бесконечности увеличивать габариты ЭВМ, поскольку это чревато и снижением их надежности, и потерями мощности. Поиск миниатюрных элементов привел к созданию микросхем на кристаллах, которые стали основой современной технологии. Сегодня их применение вышло далеко за рамки исходной цели — уменьшения габаритов ЭВМ.

Первая ЭВМ, которая умела всего-навсего быстро (для того времени) считать, была создана в 1945 г. Конечно,

современные машины не идут ни в какое сравнение со своей предшественницей по скорости счета, т. е. быстродействию. Развитие вычислительной техники от микрокалькуляторов до суперЭВМ привело к такому их разнообразию, что зачастую едва можно уловить общее в отдельных моделях. Но в каждой из них обязательно присутствует тот минимальный набор блоков, без которых устройство уже нельзя назвать ЭВМ. Изучение этих блоков необходимо для понимания принципов работы современных компьютеров.

Трудность задачи описания внутреннего и внешнего «миров» ЭВМ и их взаимосвязи, сложность восприятия и понимания возможностей ЭВМ можно проиллюстрировать, сопоставив рост возможностей человека в области энергетики и сфере информатики. Рост энергетических возможностей сопровождался удивительными примерами симбиоза зарождения идеи с ее научным обоснованием и практической реализацией: рычаг — «закон рычага»,  $E=mc^2$  — атомная энергетика. Рост же информационного обмена приводит к интенсификации умственной деятельности человека, в связи с чем ЭВМ часто называют усилителем интеллекта. С помощью ЭВМ создаются базы данных, базы знаний, экспертные системы, которые интенсифицируют процесс накопления, распространения и представления информации.

Но если легко определить, насколько усиливается мускульная энергия человека, использующего, например, рычаг, то нет даже приблизительного аналога оценки роста интеллекта в зависимости от объема информации. Очевидно только, что без информации невозможна интеллектуальная деятельность человека. В то же время это только необходимое, но далеко не достаточное условие усиления интеллектуальной деятельности человека.

# 1

## ЭВМ ВЧЕРА И СЕГОДНЯ

### 1.1. ОТКРЫТИЕ И ИЗОБРЕТЕНИЕ

Все новое, что создает человек, будь то теория или конкретное устройство, может быть отнесено к открытиям или изобретениям. В чем же разница между этими двумя понятиями?

Мы говорим, что сделано открытие, когда установлены ранее неизвестные, но объективно существующие закономерность, свойство или явление материального мира. Например, справедливость новой теоремы по сути самого ее доказательства вытекает из всей системы ранее накопленных знаний. Она как бы оставалась «незамеченной» предшественниками, но неявно присутствовала в известных им формулировках. По мере открытия новых химических элементов заполняются пустые клетки в таблице Менделеева. Сама же таблица периодической системы элементов и способ ее построения, скорее, изобретение, чем открытие. В то же время собственно периодический закон, несомненно, открытие Д. И. Менделеева.

Понятие «изобретение» можно сформулировать следующим образом. Изобретение — это новое техническое решение задачи, существенно отличающееся от ранее реализованного или не имеющее аналогов. Как правило, прикладное значение открытия осознается не сразу, в то время как изобретение непосредственно вытекает из насущных потребностей общества, проверяется его практической пригодностью, т. е. определяется социальным заказом.

История техники изобилует драматическими событиями, связанными с рождением изобретения и трудностями на пути его претворения в жизнь. На смену одному веку приходит другой, а истоки драмы остаются незыблемыми: несоответствующий уровень технологии и социальных потребностей общества ставит непреодолимые преграды распространению и реализации полезных человечеству техни-

ческих идей. Финал же этой драмы имеет классический характер. Рано или поздно добро побеждает зло, в иных социально-технических условиях идея изобретения восстает как сказочная птица Феникс из пепла и приносит многочисленные плоды.

Наиболее благоприятным обстоятельством является объективная необходимость на определенном этапе развития общества в параллельном протекании творческих процессов открытия и изобретения. Если существуют реальные предпосылки для формулировки нового теоретического положения, направленного на резкое ускорение научно-технического прогресса, а социальный заказ диктует потребность в создании новой техники и имеется достаточная для этого технологическая база, то внедрению технических новшеств обеспечена «зеленая улица». Они бурно развиваются, быстро принимаются современниками, входят в повседневную практику.

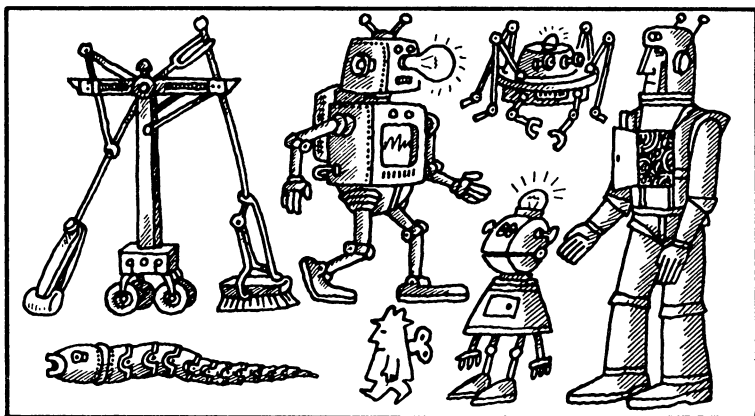
## 1.2. МЕЧТЫ И РЕАЛЬНОСТЬ

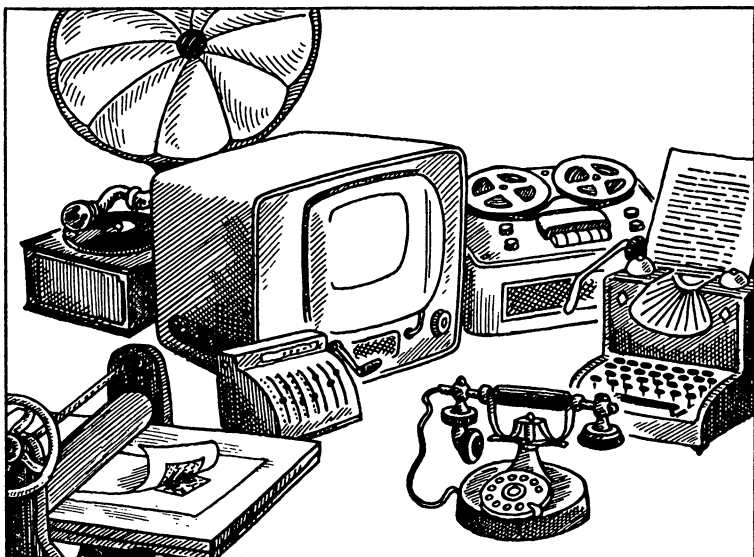
Человеку свойственно мечтать, проецировать в будущее проблемы науки, находящиеся в зачаточном состоянии. Нет числа техническим новшествам, предсказанным и предугаданным писателями-фантастами. Сколько научных прогнозов, смелых догадок и предположений, казавшихся в свое время невероятными, оправдали себя впоследствии самым непосредственным образом! Вспомним хотя бы «Наутилус» — совершеннейшую подводную лодку или летательные аппараты тяжелее воздуха, созданные прозорливой творческой фантазией Жюль Верна задолго до сколько-нибудь заметных успехов в подводном плавании и авиации.

С незапамятных времен человека влекла мечта о создании себе подобного механического помощника. В «Илиаде» Гомера бог огня и покровитель кузнечного ремесла Гефест, сын Зевса и Геры, выковывает служанок, которые действуют, повинаясь слову бога. В романе английской писательницы Мэри Шелли «Франкенштейн, или Современный Прометей» (1818) молодой швейцарский ученый Виктор Франкенштейн создает огромное человекоподобное существо, обладающее необычайной физической силой и выносливостью. Войдя в мир преисполненным благородными порывами, этот чудовищный гигант становится не «добрым волшебником», другом и помощником людей, а злобным демоном, мстящим всем и в первую очередь Франкенштейну за свое уродство и одиночество.

В XX в. широко распространилась научно-фантастическая литература, повествующая о самых разнообразных интеллектуальных автоматах. В пьесе Карела Чапека «R. U. R.» («Россумовские универсальные роботы»), написанной им в 1920 г., роботы анатомически и физиологически — полная копия человека. Они не способны только ощущать радость, играть на скрипке, любить и вообще совершать все то, что, по мнению их создателя, инженера Россума, излишне. Главный герой пьесы, президент процветающей фирмы, так излагает концепцию Россума: «Дизельный мотор не украшают резьбой и орнаментом. А производство искусственных рабочих — то же самое, что и производство дизель-моторов. Оно должно быть максимально простым, а продукт его — практически наилучшим... Россум изобрел рабочего с минимальными потребностями. Он выкинул все, что не служит непосредственным целям работы. Тем самым он выкинул человека и создал робота. Механически они совершеннее нас, они обладают невероятно сильным интеллектом, но у них нет души». Именно высокий интеллект роботов, по замыслу автора пьесы, и приводит к трагедии — уничтожению людей роботами. «За что вы нас ненавидите?» — спрашивает человек робота, на что робот отвечает: «Вы не как роботы. Не такие способные, как роботы. Роботы делают все. Вы только приказываете. Плодите лишние слова». (Кстати, К. Чапек является создателем термина «робот», который происходит от чешского слова *robota* — труд тяжелый подневольный.)

Начало и середина XX в. отмечены буквально потоком открытий и изобретений практически во всех областях нау-





*В ЭВМ использовано несколько изобретений из совершенно разных областей науки и техники*

ки и техники. Наука становится непосредственной производительной силой общества. Уже не ученые-одиночки или отдельные умельцы, а огромные коллективы занимаются решением актуальных задач усовершенствования способов труда. Стремительно развиваются авиация, ядерная физика. Математические методы в физике занимают чуть ли не доминирующее положение над экспериментальными способами. Да и сама математика, восприняв из физики новые постановки задач, в частности по расчету траекторий, уже не могла довольствоваться логарифмической линейкой.

Итак, к середине XX в. назрела потребность в автоматизации трудоемких расчетов, в создании столь сложного и дорогостоящего инструмента, как вычислительная машина. Сложнейшие расчеты не только отвлекали многочисленные людские резервы от творческой деятельности, но и зачастую были просто нереальны без ЭВМ.

Даже поверхностный взгляд на ЭВМ убеждает в том, что в ней объединены изобретения из совершенно разных областей науки и техники: телевизор и магнитофон, печать и перфорация. Из конструктивных элементов важнейшую роль играют переключательные элементы. Они характеризуются способностью удерживать два фиксирован-

ных состояния и под внешним воздействием переключаться из одного в другое. Несомненно, что до создания ЭВМ эти элементы наиболее широко применялись в технике связи, на автоматических телефонных станциях.

### 1.3. НА ПУТИ К НОВОЙ ТЕХНОЛОГИИ

Профессору математики Кембриджского университета Чарльзу Бэббиджу (1792—1871) принадлежат идея создания и проект универсальной автоматической вычислительной машины, которую он назвал аналитической. На такой разностной, или аналитической, машине Бэббиджа работала Ада Лавлейс (1816—1852), дочь поэта Джорджа Гордона Байрона, которую по праву считают первым программистом. Один из современных языков программирования — Ада назван в ее честь. Идея Ч. Бэббиджа не дала непосредственного толчка к появлению ЭВМ, но по этому направлению пошли создатели арифмометров — настольных счетно-решающих устройств.

Ч. Бэббидж использовал известные ему принципы описания информации перфорированными шаблонами. Интересно, что прообраз современных перфокарт был разработан в начале XIX в. для управления укладкой нитей в ткацких станках. Перфорированные шаблоны долгое время оставались вне поля зрения изобретателей, хотя, думается, могли быть использованы во многих устройствах и машинах. Вспомнили о них, когда стали искать способы ввода информации в ЭВМ. Суть перфорированных шаблонов (перфокарт, перфолент и др.) очень проста: наличие отверстия означает «1», отсутствие — «0». Чтобы ими воспользоваться, необходимо представить исходную информацию в кодах, содержащих два знака — 0 и 1. Затем, например, шуп или штырь (в ЭВМ — луч света) проверяет наличие отверстия, и в зависимости от того, есть отверстие или нет его, выполняется или не выполняется укладка нитей в ткацком станке (в ЭВМ — запись информационного кода).

Сегодня в вычислительной технике перфокарты и перфоленты почти полностью заменены магнитными лентами (бобинами и компакт-кассетами). Принцип хранения информации остается прежним — на ленту производится магнитная запись кодов, которые соответствуют двум состояниям: 1 — есть намагничивание, 0 — нет. Но пока присутствует медленный процесс перемотки ленты.

Основное достоинство магнитофона — возможность хранить большие объемы данных в устройстве малых размеров (по сравнению с перфокартами, например). При этом, как и в бытовом магнитофоне, от крупногабаритных бобин перешли к маленьким кассетам, и, возможно, это не предел. Кроме того, на одну и ту же магнитную ленту многократно можно записывать разные данные, в то время как перфоленты и



*Ада Лавлейс*

перфокарты предназначены для разового пользования. Благодаря небольшим габаритам магнитофонных кассет обеспечивается и более высокая скорость поиска информации. Наконец, это достаточно привычный для пользователей прибор, что является немаловажным фактором его эффективной эксплуатации.

Совершенствование магнитной записи связано с образованием в магнитном материале намагниченных областей, способных объединяться в цепи, передвигаться внутри магнитного материала. Эти области называют цилиндрическими магнитными доменами (ЦМД), а соответствующие устройства — накопителями информации на ЦМД или памятью на ЦМД. При этом виде памяти можно обходиться без механического движения, которое ограничивает скорость ввода — вывода информации. Запоминающему устройству на ЦМД свойственны высокая компактность и стабильность характеристик. Благодаря отсутствию движущихся механических частей ЦМД обладают повышенной устойчивостью к механическим воздействиям, что обеспечивает чрезвычайное удобство для использования их в системах промышленного управления, роботах и системах связи.

Прообразом АЦПУ (алфавитно-цифрового печатающего устройства) можно считать книгопечатный станок немецкого изобретателя Иоганна Гутенберга (ок. 1399—1468).

Обыкновенная пишущая машинка тоже нашла применение в вычислительной технике. Принцип ее действия сейчас является основным при вводе информации, а в пер-



вых моделях ЭВМ широко использовался и для вывода информации — клавиши букв приводились в действие программой.

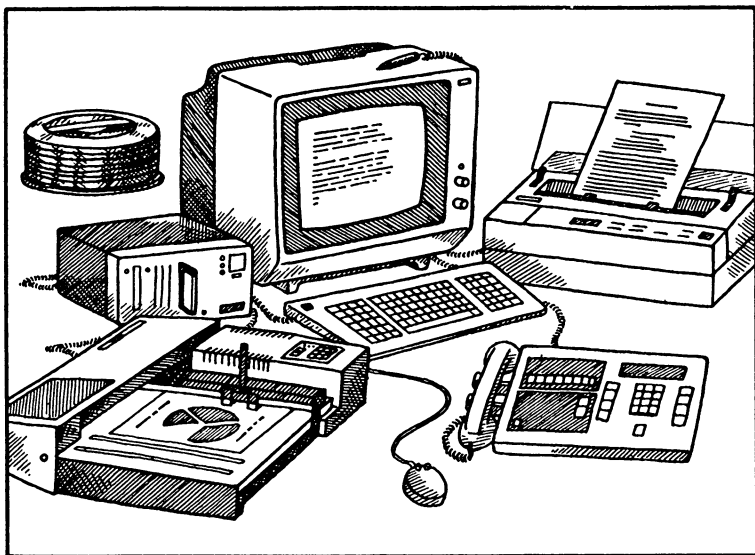
Телевизор как средство общения с ЭВМ помимо его привычности для пользователей обладает высокими скоростными характеристиками, определяемыми скоростью электронного луча. Оказав большую услугу вычислительной технике, телевизор заимствовал у нее много полезного применительно к его автономному использованию, в первую очередь — принцип дискретного представления информации (цифровые телевизоры), что существенно повысило качество изображения. Добавив небольшой блок памяти в виде кассетного магнитофона, можно воспроизводить понравившиеся передачи (видеомагнитофон) или использовать телевизор в качестве партнера в играх.

Технический прообраз еще одного из элементов ЭВМ — телефонная станция. Как известно, цель функционирования телефонной станции — коммутация каналов связи по вызову абонента. Когда телефон распространился достаточно широко, возникла потребность в надежных переключаемых схемах. А поскольку определился социальный заказ, изобретение не заставило себя ждать.

Итак, в составе ЭВМ имеется множество хорошо известных нам элементов и устройств, но ни один из них не был изобретен специально для ЭВМ; к моменту создания первой вычислительной машины все необходимые ее составляющие уже использовались на практике. Другое дело, что по мере развития вычислительной техники эти элементы подвергались модернизации. Самое значительное технологическое усовершенствование связано, несомненно, с миниатюризацией элементов памяти и арифметико-логического устройства ЭВМ, а также с уменьшением доли механического движения за счет электронного, что позволило на несколько порядков увеличить быстродействие и объем памяти (эти параметры являются определяющими при сравнении возможностей различных ЭВМ), а это в свою очередь превратило ЭВМ из вспомогательного инструмента в центральное звено автоматизированных систем.

Независимо от технических попыток создания арифметико-логических устройств проблема производства вычислений уже давно волновала ученых. В самых общих чертах суть этой проблемы сводится к постановке вопроса о том, можно ли, выполняя некоторую последовательность действий, достичь решения.

Возможно, такая формулировка показалась Вам



*При развитии вычислительной техники все составляющие элементы подверглись модернизации*

странной, но наберитесь терпения: мы еще вернемся к обсуждению этой проблемы.

Наибольший вклад в решение данной проблемы внесли американский логик Э. Пост (1879—1954), английский математик Алан Тьюринг (1912—1954) и советский ученый А. А. Марков (1903—1979). Интересно, что свою математическую модель А. Тьюринг описал в терминах технического устройства. Так называемая машина Тьюринга (1936) — это чисто математическое понятие, никоим образом не связанное с конкретным техническим решением. Оно было введено для более строгого, формального определения алгоритма — важнейшего понятия в современной вычислительной технике и являющегося одним из древнейших в математике. Вы, конечно, помните из школьной программы алгоритм Евклида (3 в. до н. э.). Но то, что достаточно понятно человеку (формализовано), не всегда однозначно понятно ЭВМ, которая не терпит многозначности.

Хотя первые технические реализации ЭВМ возникли независимо от модели Тьюринга, ее появление существенно повлияло на привлечение математиков к методологии построения ЭВМ, и это не могло не сказаться на дальнейшем пути совершенствования вычислительных машин.

Первая страница в истории создания вычислительных машин связана с именем французского философа, писателя, математика и физика Блеза Паскаля, который в 1641 г. (по другим данным в 1642) сконструировал первую суммирующую машину. Правда, она производила только сложение и вычитание, последовательно прибавляя или отнимая единицы отдельных разрядов, но механическая передача десятков была в ней принципиально осуществлена. Сохранились семь экземпляров машины Паскаля. Один из них находится в Музее искусств и ремесел в Париже, где собрана полная коллекция математических инструментов, в которую входит и модель арифмометра русского ученого П. Л. Чебышева (1821—1894).

Немецкий ученый Готфрид Вильгельм Лейбниц (1646—1716) работал над созданием универсальной машины, осуществлявшей четыре действия арифметики. Ряд важнейших ее органов применяли вплоть до середины XX в. в некоторых типах машин («Томас», «Саксония», «Архимед»). К типу машины Лейбница могут быть отнесены все машины, в частности и первые ЭВМ, производящие умножение как повторное сложение, а деление — как многократное вычитание.

Существенными частями конструкции машины Лейбница служат: установочное приспособление и подвижная каретка, в которой заключена счетная часть машины — так называемый результирующий счетчик и счетчик числа оборотов. При умножении множимое устанавливается в установочном приспособлении (на спицах или кнопках), множитель постепенно «накручивается» в счетчике оборотов и, наконец, произведение появляется в результирующем счетчике.

В дальнейшем интерес к механическим вычислителям перешел к инженерам. В 1875 г. шведским инженером В. Т. Однером, работавшим в России, была сконструирована машина, которая впоследствии получила наименование арифмометра. Герман Холлериз для проведения переписи населения США на 1890 г. сконструировал машину, автоматизировавшую процесс обработки данных, и использовал в качестве носителей информации перфокарты. В 1896 г. он основал фирму по выпуску перфокарт и счетно-перфорационных машин. В дальнейшем она была преобразована в известную фирму — производитель вычислительной техники — IBM.

Появление ЭВМ в середине XX в.— вполне закономерное событие. Тем не менее, видимо, оно застало врасплох тех, кто назвал этот новый инструмент электронной вычислительной машиной. Какая же она электронная, если есть и другие технологические решения? Почему вычислительная, если существуют машины, выполняющие только поиск информации или управление? Пожалуй, бесспорным в названии является слово машина, хотя оно мало что поясняет из-за обширности класса машин и их разнообразия. На английском языке ЭВМ именуется просто «вычислитель» (computer — компьютер), чтобы отличать ее от счетных устройств — калькуляторов, а на французском — «упорядочивающий» (l'ordinateur), чем акцентируется роль программы, а не функциональное назначение. Этот разнобой в названиях свидетельствует прежде всего о широте возможностей применения и разнообразии технического воплощения ЭВМ. Но чтобы сориентироваться в этом разнообразии, попробуем рассмотреть самую простую ЭВМ, не обладающую всеми многогранными способностями современных машин, но по праву относящуюся к их семейству.

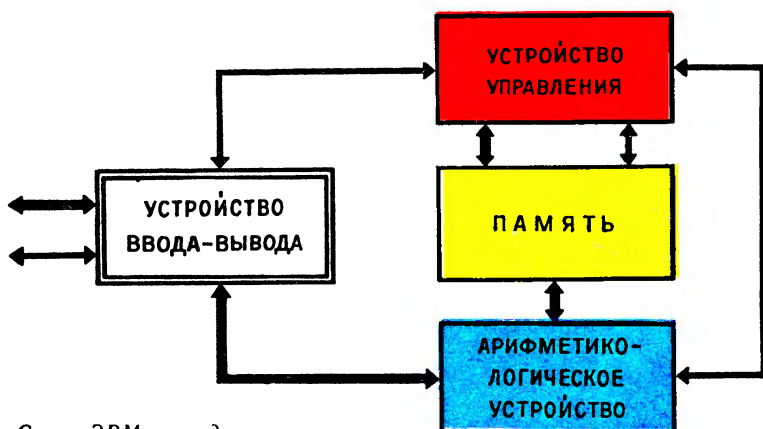
В 1925 г. В. Буш создает первую вычислительную машину на электрических реле. Это был новый шаг в технологии, но не в автоматизации. Управление процессом счета все еще возлагалось на человека. В 1944 г. появилась релейная машина «Марк-1», в которой управление вычислениями осуществлялось автоматически. Новый элемент этой машины — устройство управления. Его роль выполняло зубчатое колесо, которое перематывало управляющую перфоленду. Но, как и всякое механическое устройство, машина «Марк-1» не обладала тем быстродействием, которое позволило бы осуществить качественный скачок в технологии вычислений.

Первая быстродействующая электронная вычислительная машина ЭНИАК, созданная американскими специалистами в Пенсильванском университете, имела в своем составе 18 тыс. электронных ламп, потребляла более 100 кВт электроэнергии, весила 30 т и занимала комнату длиной 30 м. Сложение и вычитание производились за 200 мкс (в 1000 раз быстрее, чем в машине «Марк-1»), умножение — за 2300 мкс. ЭНИАК предназначалась для решения дифференциальных уравнений в задачах расчета траекторий, т. е. была специализированной.

(С 1948 г. начинается серийный выпуск универсальных ЭВМ фирмой IBM, которая и до настоящего времени является крупнейшим производителем вычислительных машин.)

Машина ЭНИАК была рассчитана на выполнение конкретной последовательности вычислений. Если требовалась иная последовательность, схему приходилось практически заново монтировать. Проект первой ЭВМ заинтересовал известного математика Джона фон Неймана (1903—1957), и он занялся разработкой логической схемы, способной использовать гибко запоминаемую программу, которую можно было бы изменять, не перестраивая всей схемы машины. Как следовало из сообщения Дж. фон Неймана по этому поводу в 1946 г., он впервые выделил четыре основных блока в схеме ЭВМ и назвал их арифметико-логическим устройством, памятью для данных и команд, устройством управления и устройством ввода — вывода. Впоследствии ЭВМ такой конфигурации получила название машины неймановского типа. Реализовать собственные принципы на конкретной ЭВМ Нейману удалось только в 1952 г. Основной его заслугой является разработка принципа гибкого программного управления в последовательных машинах с вычислительной программой.

Первая отечественная и первая в Европе вычислительная машина МЭСМ (малая электронная счетная машина) была разработана в 1950 г. под руководством академика С. А. Лебедева (1902—1974). МЭСМ имеет более универсальное назначение, нежели ЭНИАК, и по существу структурные схемы этой машины являются классическими



*Схема ЭВМ последовательного типа*

(они положены в основу последующей серии отечественных ЭВМ — БЭСМ). Она могла обрабатывать 50 операций в секунду и хранить в оперативной памяти 31 число и 63 команды. Уже в первой ее модели был заложен прообраз внешней памяти емкостью 5000 слов на магнитном барабане. Хотя МЭСМ проектировали, скорее, с целью отработки конкретных научных и технических идей, они были использованы и практически для решения народнохозяйственных задач. Эта машина сыграла большую роль в качестве макета, на котором отрабатывали принципы построения семейства БЭСМ, характеристики которых не уступали лучшим зарубежным моделям.

Первая БЭСМ появилась в 1952 г. Ее быстродействие составляло уже  $10^5$  операций в секунду, а объем оперативной памяти достиг 1024 слова. Внешняя память на двух магнитных барабанах объемом 5120 слов каждый и на магнитной ленте емкостью 120 тыс. чисел обеспечивала решение сложных задач.

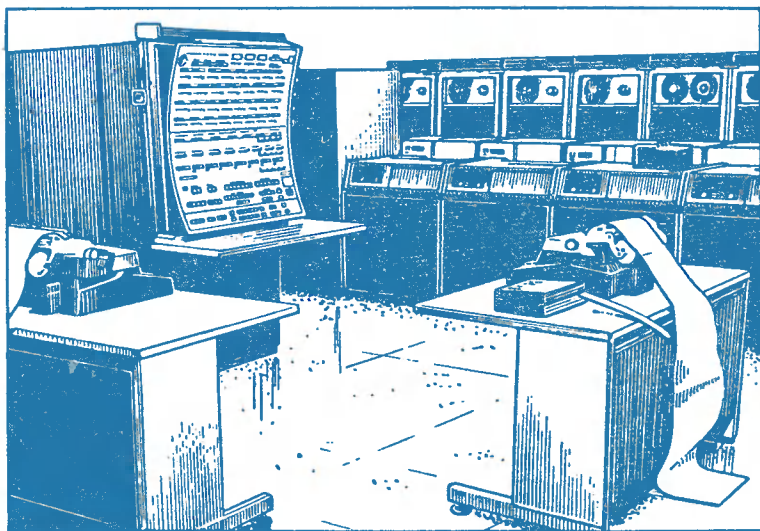
В 1967 г. была создана самая мощная ЭВМ этого семейства — БЭСМ-6 с быстродействием 1 млн операций в секунду. Время выполнения операции сложения — 1,2 мкс, умножения — 2,1 мкс. Емкость оперативной памяти — 32 тыс. слов.

В этот период развитие вычислительной техники шло по пути наращивания мощностей всех основных блоков — памяти, арифметико-логического устройства, устройства управления. Совершенствовались и устройства ввода — вывода. Машины имели универсальное назначение. Правда, то, что подразумевали под универсальностью 20 лет назад, в наши дни уже на заслужило бы такого названия. Например, ЕС ЭВМ и сегодня называют универсальными, хотя уже свыше 10 лет существуют управляющие мини-ЭВМ (например, М-6000), которые имеют более широкий спектр применения, включая традиционные вычислительные задачи. Дальнейшее развитие в отдельности каждого из таких блоков, как память, арифметико-логическое устройство, устройство управления и устройство ввода — вывода, привело к появлению специализированных ЭВМ.

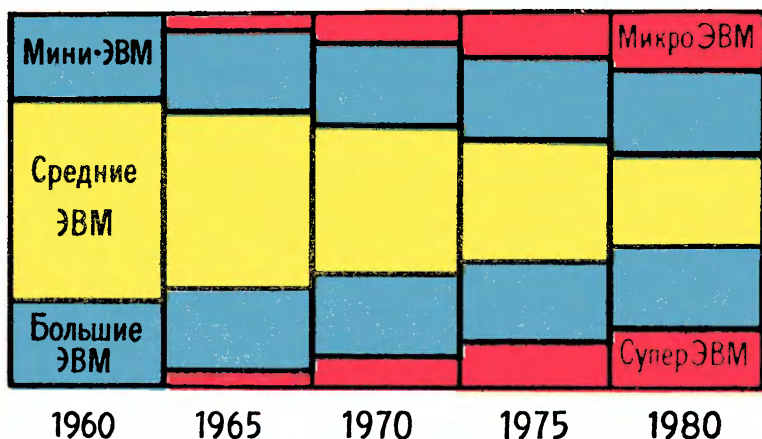
С позиций технологии ЭВМ за 40 лет претерпели серьезные изменения, о чем свидетельствует переход от электровакуумных приборов, транзисторов к микросхемам, большим интегральным схемам, кристаллам... Но важно, что при любом технологическом исполнении все основные блоки выполняют те же самые функции, хотя и с различными скоростью и надежностью.

## 1.6. ЭВМ СЕГОДНЯ

К 70-м годам был накоплен большой опыт использования ЭВМ. С появлением новых областей их применения пользователи, как правило, нуждались в какой-то одной функции ЭВМ и недоиспользовали другие. Все это привело к созданию специализированных ЭВМ. И хотя в специализированных ЭВМ все основные блоки присутствуют, один из них как бы доминирует над остальными. Например, в автоматизированных системах хранения информации самое мощное звено — память, в так называемых управляющих ЭВМ — блок управления, в быстродействующих вычислительных машинах — арифметико-логическое устройство. Что касается устройств ввода — вывода, то именно прогресс в разработке новых способов и средств взаимодействия с ЭВМ — от традиционных принтеров и графопостроителей до манипуляторов, роботов и станков с числовым программным управлением — приводит к широкому разнообразию и самих ЭВМ. Недаром роботы, способные подобно человеку рисовать картины и перемещаться на двух ногах, обычно являются демонстрационными экспонатами предприятий, специализирующихся в области создания вычислительной техники. Все «интеллектуальное» в роботе, станке и других системах основано на их совместной, комплексной



*ЭВМ большой производительности ЕС-1060*



*Соотношение в использовании ЭВМ разной мощности*

работе с ЭВМ, управляющей и координирующей работу исполнительных органов.

По быстродействию (числу элементарных арифметических или логических операций, выполняемых за секунду) современные ЭВМ делятся на несколько классов. Так, суперЭВМ способны осуществлять сотни миллионов операций в секунду, большие ЭВМ — до 10 млн. операций, средние ЭВМ — до 1 млн., а малые ЭВМ, персональные компьютеры и встроенные микропроцессорные устройства — только десятки или сотни тысяч операций. Потребности в ЭВМ разных классов разнятся. Даже самым высокоразвитым странам нужны единицы суперЭВМ. Больших ЭВМ необходимо сотни — они составляют основу крупных отраслевых и региональных вычислительных центров. А вот малых, персональных ЭВМ и микропроцессорных управляющих устройств требуются миллионы — в будущем ими должны быть оснащены каждая производственная установка и любой технологический узел, каждое рабочее место инженера и научного работника.

Непрерывное повышение требований к вычислительной мощности средств обработки информации обуславливает необходимость решения проблемы, связанной с достижением рекордных, практически предельных для ЭВМ скоростей.

Сегодня суперЭВМ — это самые большие, самые быстродействующие, наиболее сложные и дорогостоящие ЭВМ, предназначенные для решения сверхсложных научно-технических проблем, что диктует потребность в крайне



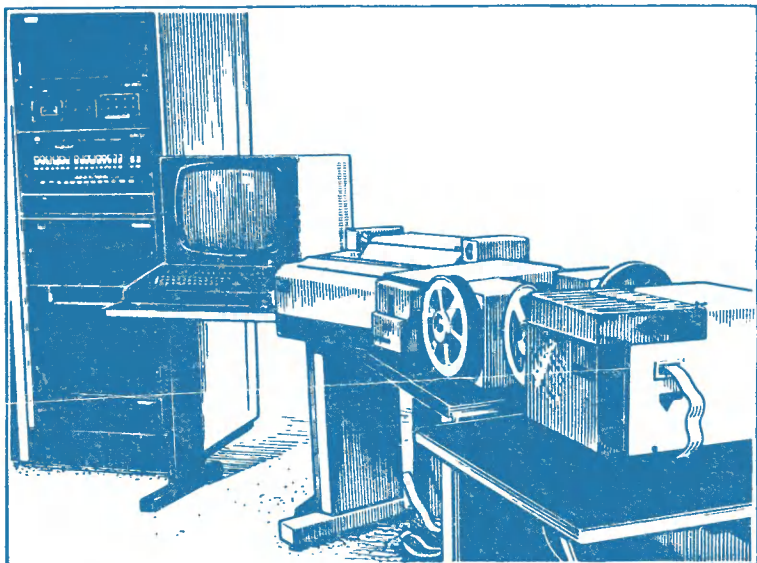
большом объеме вычислений. СуперЭВМ — представитель сравнительно нового, весьма динамично развивающегося класса сверхмощных машин. Хотя численность этих машин сейчас сравнительно невелика, именно на них приходится заметная часть суммарной производительности и общей стоимости всего парка ЭВМ. Разработчики и проектировщики уже говорят о создании супермини-ЭВМ и супермикроЭВМ — самых совершенных вычислительных машин, в которых высокие мощности сочетаются с высокой степенью миниатюризации.

Термин «суперкомпьютер» применительно к сверхбыстрым ЭВМ начал входить в обиход в середине 60-х годов, однако первые действующие суперкомпьютеры появились лишь в начале 70-х годов. В настоящее время на повестке дня уже стоит вопрос о необходимости создания машин, более чем в 100 раз превосходящих по вычислительным возможностям лучшие современные суперЭВМ. Непрерывное улучшение технологии изготовления переключательных схем сделало возможным работу в субнаносекундном диапазоне. Мы приближаемся к теоретическому пределу быстрогодействия, равнозначному скорости света. Пока же рекорд производительности принадлежит суперкомпьютеру, осуществляющему 800 млн. операций в секунду.

СуперЭВМ в основном используют сейчас при решении задач численной обработки данных и в сетях ЭВМ коллективного пользования, где наиболее полно реализуются ее огромные мощности как координатора работы всех включенных в сеть больших и малых машин.

Мини-ЭВМ быстро нашли свое место в лабораториях. Ученые по достоинству оценили эти машины, ориентированные на решение определенного класса задач. Вместо того чтобы выполнять задачи различных типов на универсальной машине-гиганте, пользователь получил возможность переходить от одной машины к другой, ориентированной на иной класс задач. Сама идея создания устройства, имеющего архитектуру ЭВМ и предоставляющего пользователю монополию выполнения одного задания, оказала существенное влияние на дальнейшее развитие вычислительной техники. Однако до тех пор, пока не появились дешевые мини-ЭВМ, мало кто мог позволить себе иметь машину только для решения индивидуальных задач сравнительно узкого класса.

Характеристики мини-машины значительно приблизились к характеристикам больших машин. Они обладают высоким быстродействием (цикл обращения к оперативной памяти менее 1 мкс). Емкость оперативной памяти



*Мини-ЭВМ СМ-1*

мини-машины уже достигла нескольких мегабайтов. Расширенный список команд (400 и более) позволяет осуществлять эффективную обработку информации.

За короткий срок, буквально на наших глазах, изменился подход к ЭВМ и возможностям их использования. Вспомним, что ЭВМ вначале были созданы для выполнения сложных расчетов. Но вскоре выяснилось, что они могут служить незаменимым средством при решении информационных и управленческих задач.

С каждым годом вычислительные машины, содержащие значительное число микропроцессоров, становятся все более грозными конкурентами «обычных» больших машин. Результаты исследований показывают, что машина, содержащая 10—20 микропроцессоров, взаимодействующих с мини-машиной, сможет решать задачи одинаковой сложности в десятки раз быстрее. Стоимость же ее будет неизменно уменьшаться. Рассматривается даже возможность создания машины, содержащей около 10 тыс. микропроцессоров.

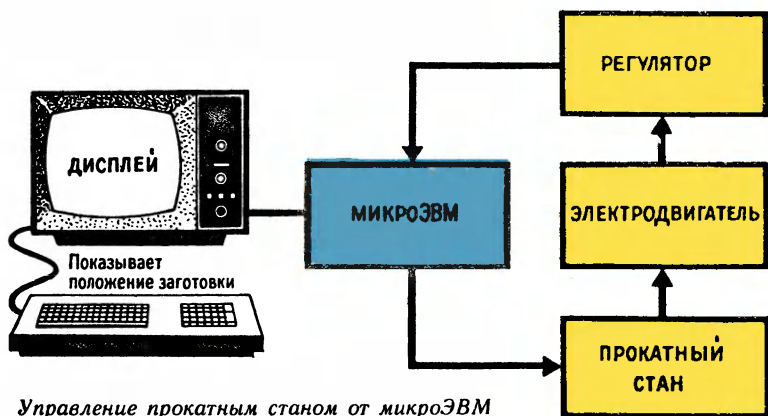
Успехи широкого применения в вычислительной технике микропроцессоров оказались настолько существенными, что даже возник вопрос: нужны ли крупные машины?

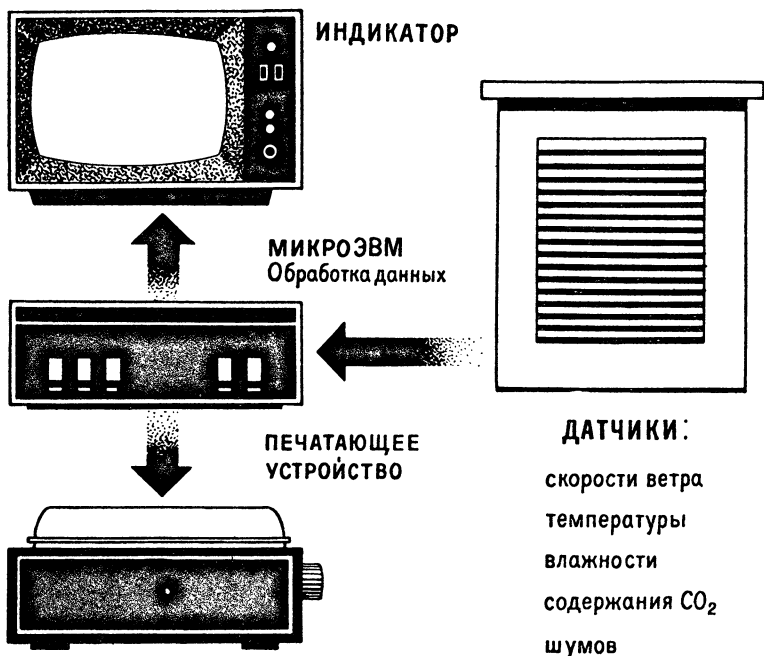
Со временем м и к р о Э В М полностью заменят ЭВМ во всех технологических узлах. Примерами использования

микроЭВМ для целей управления являются металлорежущие станки с числовым программным управлением (ЧПУ), сборочные автоматы, разнообразные установки по изготовлению и обработке деталей, регистраторы данных, устройства интерфейса и терминальные устройства ЭВМ. МикроЭВМ внедряются и в сферу быта. Их используют в электронных устройствах, расширяющих функции наручных часов, в почтовых автоматах, регуляторах оптимального расхода горючего в автомобиле и т. д. В системе автоматического управления прокатным станом микроЭВМ предварительно определяет положение заготовки, подлежащей прокату. Далее на основе произведенной оценки с учетом скорости вращения электродвигателя, соединенного передаточным механизмом с прокатным валиком, микроЭВМ определяет разницу между требуемой скоростью вращения и действительной и вырабатывает сигнал для регулирования этой скорости. Такая система обеспечивает высокую точность и надежность регулирования при довольно низкой стоимости.

Примерами использования микроЭВМ для целей измерения являются автоматические индикаторы в электронной аппаратуре, измерительно-регистрирующие устройства контроля, телеметрические установки на станциях, тестеры для интегральных схем, устройства для измерения и контроля скорости автомобиля и т. д. При измерении информация о различных физических величинах поступает в микроЭВМ с датчиков сигналов.

Быстрое снижение стоимости микропроцессоров привело к появлению нового типа машины — персональной ЭВМ, предназначенной для использования в научной ра-



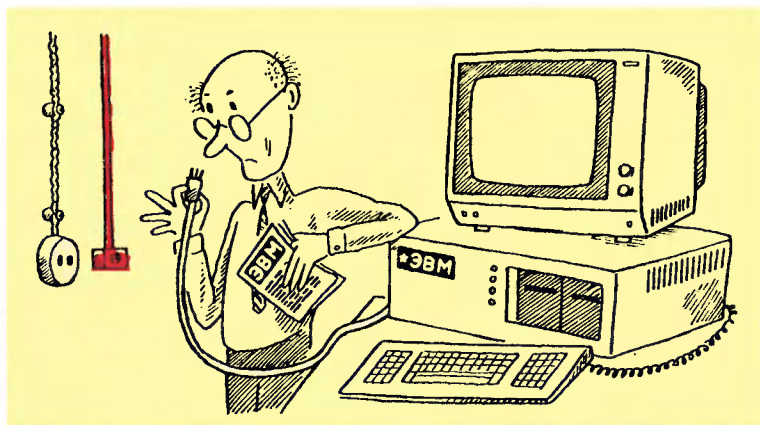


*Сбор и обработка метеоданных с помощью микроЭВМ*

боте, служебной деятельности или домашнем хозяйстве. В последнем случае особенно важно, что в роли внешних устройств могут выступать обычные бытовые аппараты: телевизор, магнитофон, телефон. Стоимость персонального компьютера, как правило, невысока. Например, стоимость ЭВМ БК-0010 примерно равна стоимости большого черно-белого телевизора. Широкое распространение получил персональный компьютер при организации досуга — это и телеигры, и синтезаторы звука, и электронные инструменты.

В более сложных деловых играх персональные ЭВМ постепенно вытесняют большие и мини-ЭВМ. Кстати, с создания игр и началось проникновение персонального компьютера в область обучающих машин.

На нынешнем этапе развития экономики страны, когда связи между различными организациями усложняются, а динамика развития народного хозяйства требует четкости в управлении, быстрого принятия решений, особую значимость приобретает такая новая отрасль знаний, как информатика. В ее задачи входит разработка методов и средств,



обеспечивающих высокую скорость сбора, передачи, обработки и выдачи данных, причем в форме, удобной для работников производства и аппарата управления. Персональный компьютер, подключенный к распределенной информационно-вычислительной сети, становится таким же обязательным элементом структурной организации взаимодействия во всех сферах человеческой деятельности, как и телефон.

## 2 ЧТО ТАКОЕ ЭВМ?

### 2.1. ПАМЯТЬ

В массовых периодических изданиях (газетах, журналах) и в литературе, в частности художественной, так часто речь идет о вычислительных машинах, что, вероятно, какой-то образ ЭВМ сложился у любого человека, даже далекого от вычислительной техники. Писатели-фантасты нередко описывают некие человекоподобные автоматы, которые не так-то просто отличить от человека. И хотя не хочется рассеивать определенные иллюзии, возможно, возникшие на этот счет у некоторых читателей, все же мы вынуждены откровенно признать, что ЭВМ сегодня гораздо больше схожа с любой другой машиной, нежели с человеком.

Чтобы ЭВМ стала верным помощником, очень важно понимать основные принципы ее действия. Если наш читатель, прочитав эту книгу, обратится к более детальному и серьезному рассмотрению вопросов, связанных с ЭВМ, то он, несомненно, найдет в вычислительной машине и «учителя», и «партнера в играх», и удобный и надежный инструмент в своей будущей профессии.

Мы уже узнали, что в первых ЭВМ, как, впрочем, и в большинстве современных, присутствуют четыре основных блока: память, арифметико-логическое устройство, блок управления и блок ввода — вывода. Чтобы осознать важность и необходимость этих основных частей, попробуем проанализировать собственные действия в простых ситуациях.

Приступая к решению арифметической задачи, сначала Вы читаете условия задачи и тем самым вводите в свой «индивидуальный компьютер» — головной мозг — информацию о том, какие действия и над какими числами нужно произвести. Затем Вы размышляете над тем, каким образом можно решить задачу, или, если задача часто встре-

чалась раньше, вспоминаете готовый ответ. Иногда вспомнить очень легко, например, отвечая на вопрос «сколько будет дважды два?», так как часто используемые данные хранятся в оперативной памяти. Но порой придется поломать голову, а в ряде случаев и порыться в книгах. Примерно так же обстоит дело и в ЭВМ: либо в памяти ее хранятся готовые ответы, либо в арифметико-логическом устройстве осуществляется вычисление. И если координирует последовательность и согласованность всех описанных действий человека центральная нервная система, то в машине аналогичные функции выполняет устройство управления. Наконец, получив ответ, Вы его произносите или записываете, а в машине для этих целей предусмотрен блок ввода — вывода.

Память человека — это сложнейшее явление, еще мало изученное с позиций и механизма запоминания, и размещения запоминаемой информации. В момент запоминания или воспоминания подключаются способности человека к обобщению и детализации знаний, к сопоставлению разрозненных понятий, к зрительному, слуховому, логическому восприятию информации. Ученые пока еще далеки от воспроизведения искусственными способами нашей удивительной способности использовать память во всех ее проявлениях.

Когда мы говорим о памяти ЭВМ, то имеем в виду нечто гораздо более примитивное, нежели память человека. Скорее, она напоминает некую емкость, разделенную на ячейки. Вы заполняете и надписываете каждую ячейку. Затем, просматривая надписи, открываете ту ячейку, содержимое которой Вас интересует. Примерно такого рода память имели первые ЭВМ. Дальнейшее развитие и усложнение памяти ЭВМ шло по пути хотя бы частичного воплощения в ней свойств памяти человека. Чтобы осознать важность предпринимаемых в этом направлении попыток, нужно рассматривать память неразрывно с процессом поиска.

Часто ли нам приходится что-либо искать? Даже при полном порядке в доме всякое дело сопровождается поиском каких-либо предметов, записей... Сам порядок прежде всего и предполагает возможность быстро находить нужное. В вычислительной машине поиск оказывается настолько важным, что затратами времени на поиск измеряют производительность ЭВМ. С одной стороны, это объясняется тем, что машина исполняет записанные в программе действия последовательно одно за другим и каждое действие требует небольшого числа операндов (объектов, над которыми это действие производится). Ведь и читатель в

библиотеке, имея большой список книг для прочтения, может одновременно читать только одну книгу. С другой стороны, мы хотим, чтобы записанные действия относились к большому количеству объектов, т. е. могли быть повторены для другой группы операндов.

Чтобы получить в библиотеке нужную книгу, необходимо знать фамилию автора, название книги, иногда год ее издания и издательство, выпустившее эту книгу. Точно так же и объекты для программных действий имеют свои признаки, позволяющие отличать один объект от другого. Этими признаками являются либо имена, либо числа, которые так или иначе количественно характеризуют объекты. Существуют и относительные признаки, которые сопоставляются с некоторыми шкалами, например: цвет (от красного до голубого), возраст (от младенчества до старости) или просто место в какой-либо очереди (первое, последнее).

Все действия, с помощью которых по определенным признакам ЭВМ находит объект среди ему подобных, называют процедурой поиска или просто поиском.

Приведем типичный пример. На проезжей части был сбит человек. Водитель, пользуясь темнотой, попытался скрыться. Работник ГАИ в результате опроса очевидцев происшествия установил, что наезд был совершен на красном «Запорожце» с номерным знаком 98-45, за рулем которого был молодой мужчина. Из-за отсутствия серии требуется сложная процедура поиска в картотеке. ЭВМ последовательно из памяти выдает возможные адреса владельцев автомашины. Однако их может быть очень много, и опытный инспектор откорректирует запрос, выявив прежде всего адреса автолюбителей, проживающих неподалеку от места происшествия, или расширив диапазон поиска, так как в темноте точно распознать цвет автомобиля затруднительно. В задачах такого рода ЭВМ — незаменимый помощник благодаря способности быстро проанализировать большой объем информации.

Разработчики ЭВМ выделили самый главный признак, которым обладает любой объект, — его место среди других. Этот признак называли адресом. Всю память машины разделили на небольшие части, и каждой из них присвоили свой адрес. Но адрес в привычном, бытовом смысле, т. е. название города и улицы, номер дома и квартиры, довольно сложен для ЭВМ. Машинный адрес памяти — это только число, причем если разделить память всего на 1000 частей, то адрес памяти будет иметь значения от 0 до 999. Те части памяти, которые адресуются, принято называть ячейками.



Процедуру поиска по адресу сделали очень быстрой — продолжительность ее около 100 наносекунд \* — и назвали адресными командами чтения и записи, а память, в которой они работают, — оперативной.

В современных ЭВМ существуют три вида памяти: оперативная, внешняя и долговременная. Суммарный объем памяти огромен, но существенное ограничение накладывается на оперативную (до 10 Мбайт \*\* на один процессор) и внешнюю память (около 10 Гбайт), поскольку увеличение их емкости связано со значительными денежными затратами, а стоимость ЭВМ является немаловажным фактором в экономике. Конечно, время поиска для этих видов памяти меняется ступенчато: при подключении внешней памяти на это требуются уже микросекунды, а долговременной памяти — единицы и десятки секунд.

Необходимость перехода от одного вида памяти к другому обусловлена не только временем поиска, но и различием в принципах действия всех трех видов памяти. Оперативная память опирается на электрические свойства полупроводниковых материалов, внешняя память организуется в большинстве случаев на базе магнитной записи. Долговременная память при высокой компактности обеспечивает возможность длительного хранения информации, для чего используют микрофиши (вариант микрофильмирования), пластинки с голографической записью, диски с лазерной цифровой записью.

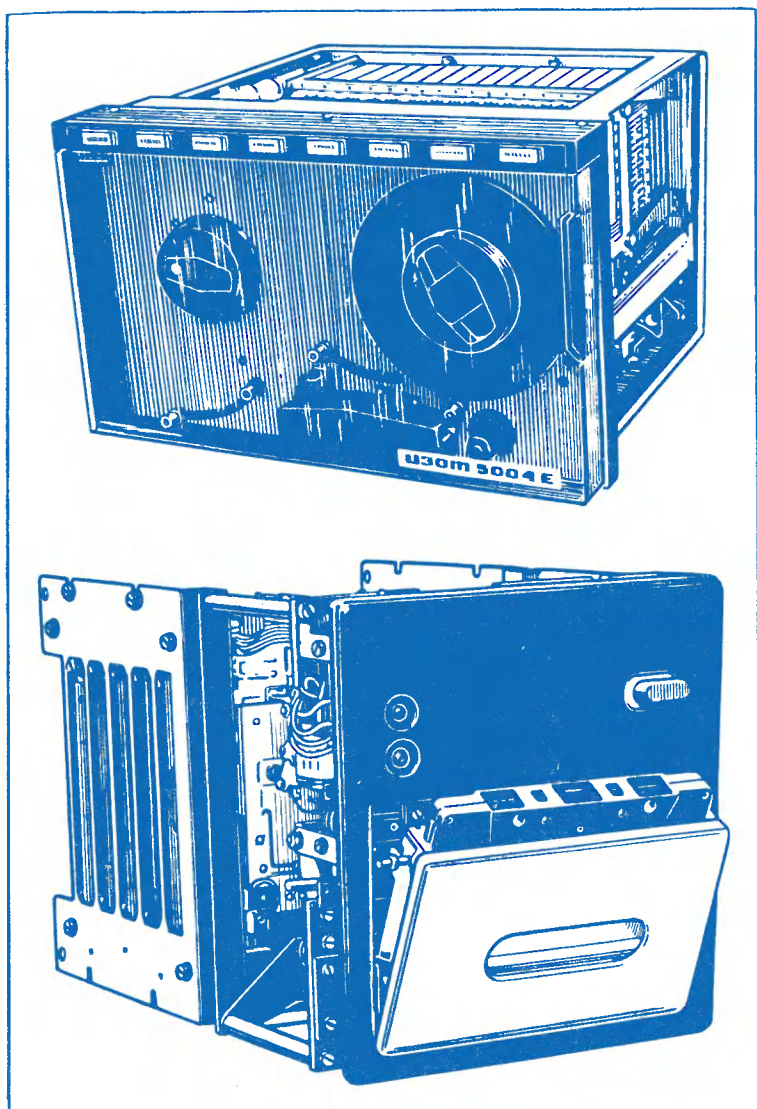
Целесообразность ступенчатой организации памяти ЭВМ легко подтвердить примерами, заимствованными из повседневной жизни. На книжных полках библиотеки Вы легко и быстро найдете, скажем, роман Л. Н. Толстого «Война и мир». Заказав же книги редко запрашиваемые, нужно запастись терпением. А на поиски и доставку уникальных изданий XVII—XVIII вв. потребуются несколько дней. Итак, разница во времени поиска существенна.

Интересно, что ступени памяти наблюдаются и у человека. «Оперативная» память позволяет моментально отвечать на вопросы «как Вас зовут?», «сколько Вам лет?». Если же Вас спросят, «как зовут человека, заходившего к Вам в позапрошлый четверг?», то, вероятно, придется со-

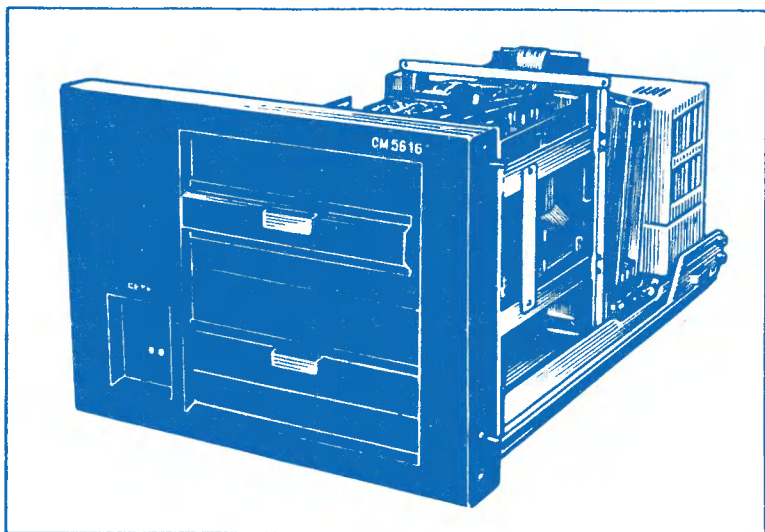
---

\* 1 наносекунда =  $10^{-9}$  с.

\*\* В соответствии с СИ употребляются приставки: кбайт — килобайт —  $10^3$  байт, Мбайт — мегабайт —  $10^6$  байт, Гбайт — гигабайт —  $10^9$  байт. Обычно большие размеры памяти характеризуют в этих единицах округленно. Например, память 4096 байт  $\approx$  4 кбайт, 4 194 304 байт  $\approx$  4 Мбайт.



*Внешняя память на магнитных лентах*



#### *Внешняя память на магнитных дисках*

средоточиться, чтобы выделить необходимую информацию, а это потребует больше времени. Ну, а вопрос «какие животные водятся в Гималаях?» скорее всего заставит Вас обратиться к книгам и, в свою очередь, вспомнить, к каким именно. Очень похоже на оперативную, внешнюю и долговременную память! Впрочем, очевидно, некоторые из наших читателей на последний вопрос моментально дадут ответ. Но это лишний раз свидетельствует о том, что в «оперативной» памяти находится именно та информация, к которой наиболее часто обращаются, т. е. здесь проявляется специализация знаний при профессиональной ориентации. Таким свойством обладает и ЭВМ. Об этом стоит поговорить отдельно. Ведь от того, как распределена информация, существенно зависит эффективность работы как человека, так и ЭВМ.

Итак, память распределяется по разным уровням таким образом, что каждый из уровней содержит набор однородных по информационной емкости ячеек. Что же такое ячейка памяти? Проще всего представить ячейку в виде цепочки связанных между собой переключательных элементов. Обыкновенная лампа включается переключательным элементом, в котором могут быть только два состояния: есть напряжение и нет его. Подобные элементы составляют и ячейку ЭВМ. Предположим, нужно хранить в ячейке чис-

ло 1001. Будем считать, что «1» соответствует высокому напряжению, а «0» — низкому напряжению или его отсутствию. Тогда на два крайних переключательных элемента нужно подать высокое напряжение, а на два средних — низкое.

В самых первых ЭВМ в качестве переключательных элементов использовали электровакуумные лампы, затем стали применять элементы полупроводниковые, а сейчас — в виде микросхем. Употребление микросхем вместо отдельных элементов (триодов, диодов, резисторов и т. д.) позволило существенно изменить конструкцию запоминающего устройства, сделать его компактнее и надежнее. Сегодня редко выполняют внешнюю память на полупроводниковой основе. Полупроводники используют только в тех случаях, когда требуется очень большое быстродействие при работе с большими массивами данных (так называемая кэш-память). Но суть действия осталась прежней: фиксировать одно из двух состояний электрического или магнитного поля. Сейчас прогресс в технологии в основном направлен на повышение надежности элементов памяти и уменьшение их габаритов. Значительно чаще для внешней памяти употребляют магнитные ленты (бобины, кассеты) и диски. Постепенно ленты и диски вытесняются устройствами памяти на цилиндрических магнитных доменах (ЦМД).

Расположенные рядом элементы в микросхеме или области намагниченности на ленте называют ячейкой памяти для хранения одного машинного слова или его части. Длина слова почти всегда \* кратна байту — единице количества информации в Международной системе (СИ). Один байт состоит из восьми бинарных (двоичных) элементов, способных к четкому (стабильному) удержанию двух противоположных состояний. При этом бинарный элемент содержит информацию в 1 бит.

Понятие «ячейка» относится к конструкции памяти. Ячейка состоит из разрядов (бинарных элементов памяти), и ее длину измеряют количеством разрядов. Иногда говорят, что ячейка содержит, например, 4 байта.

Итак, пусть в память ЭВМ заложена информация в 64 кбайта, состоящая из слов по 2 байта, тогда в ячейке 16 разрядов, а всего в памяти 32 тыс. ячеек.

---

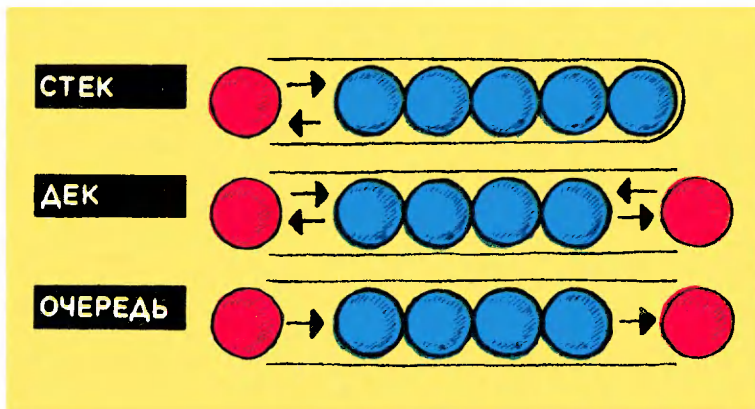
\* Исключение составляет память специализированных ЭВМ, но и в этих случаях с целью обеспечения технологичности устройства стремятся, чтобы длины машинных слов были кратны байту.

Не догнав память человека по сложности выполняемых функций, ЭВМ в какой-то степени компенсирует это отставание за счет специализации больших объемов памяти.

В зависимости от того, какой круг задач будет выполнять конкретная ЭВМ, проектируется и определенная конфигурация памяти. При вычислениях мы имеем дело с числами, которые несложно упорядочить. При решении задачи хранения неколичественной информации следует сгруппировать ее таким образом, чтобы близкая по смыслу информация находилась рядом, иначе при ее поиске пришлось бы все время пролистывать страницы долговременной памяти. Проблема удобного для последующей работы размещения данных решена сейчас лишь для ограниченного числа приложений ЭВМ. Например, ЭВМ получили широкое распространение в библиотеках как массовых, так и научных, в хранилищах технических чертежей и даже в музеях. Поскольку при этом требуется очень большой объем памяти, используют, как правило, долговременную память. Этот вид памяти по способам записи информации ближе всего к привычному для нас тексту и отличается от него в основном габаритами. Так, на микрофишу — фотографическую пластинку размерами чуть меньше почтового конверта — помещаются 72 страницы текста. Собственно, почтовый конверт и определил размеры микрофиш, поскольку при их изначальном применении в библиотечном деле отснятые копии журналов, книг пересылались по почте. Голографический способ позволил изменить форму и компактность долговременной памяти, но не ее суть: мы храним текстовые данные, не кодируя их как-либо дополнительно.

Все же есть и некоторые общие принципы организации памяти для решения различных задач. Поскольку память мы рассматриваем как последовательность ячеек, то с целью ускорения поиска имеет смысл выделить специальные адреса, в которые новые данные записываются, и адреса, из которых они могут быть прочитаны. Расположение этих групп адресов и перемещение данных между ними определяют тип организации доступа к данным в памяти.

Рассмотрим три варианта организации доступа. Если в одну область адресов данные только записываются, из другой только прочитываются и данные последовательно перемещаются от места записи к месту чтения, то такой вид памяти принято называть очередью, что вполне соответствует общепринятому значению этого слова. Краткая формулировка этого способа организации памяти такова:



### *Способы чтения и записи в памяти ЭВМ*

«первым пришел — первым ушел». Для работы с такой памятью требуются всего три операции: чтение, запись и продвижение очереди.

Если же место чтения и место записи данных совместить, то получим стек. Тогда операции чтения и записи производятся из одной и той же ячейки, причем читается то данное, которое было записано последним. Кратко сформулировать этот способ организации памяти можно следующим образом: «последним пришел — первым ушел». Продвижение данных в этом случае схоже с перемещением монет в монетнице.

Если при первом способе чтение и запись вести с обоих концов или, что то же самое, открыть второй конец стека для чтения и записи, то получим дек. Дек по организации доступа напоминает вагон с дверьми на концах — в каждую дверь можно входить и из каждой выходить.

При работе с деком можно использовать свойства очереди и стека: при накоплении данных продвигать очередь от одного конца к другому и одновременно считывать данные с того конца, с которого они были записаны.

Из этих способов наиболее употребим стек. Как единственный железнодорожный станционный путь, или тупик, позволяет сортировать вагоны и формировать составы, так и в работе с памятью ЭВМ стека вполне достаточно для разнообразных действий с данными. Наибольшая скорость операций чтения и записи данных достигнута в специальной сверхоперативной стековой памяти. Оснащение ЭВМ таким стеком существенно повышает ее производительность.

Люди научились считать еще в глубокой древности. Причем в качестве первого счетного прибора служили десять пальцев. Постепенно усложнялись операции счета. Для облегчения вычислений изобретали различные приспособления — счеты, арифмометры, логарифмические линейки. Наконец, появился очень мощный инструмент счета — ЭВМ.

Способность к быстрому счету у людей встречается крайне редко — это исключительное явление. Они и сами объяснить не могут, как производят вычисления; обычно говорят, что сразу видят ответ или что ответ возникает сам собой. Что же касается ЭВМ, то для них быстрый счет не только норма, но и основа многих процессов, внешне не связанных со счетом (например, поиск информации). Знание того, как считает машина, позволяет достичь увеличения ее быстродействия. Для этих целей, например, организуют так называемые параллельные вычисления, в ходе которых одновременно обрабатываются несколько чисел или несколько разрядов одного числа.

Одно из первых направлений распараллеливания процессов в ЭВМ заключалось в совмещении по времени этапов выполнения соседних операций. Идея совмещения операций для повышения скорости вычислительных машин принадлежит академику С. А. Лебедеву. В 1957 г. он предложил принцип совмещения операций, при реализации которого для вызова очередной команды не требовалось дожидаться отсылки результата выполнения предыдущей команды. Арифметические действия стали выполняться одновременно с обращением к памяти.

Чаще машина осуществляет действия над двумя и даже одним числом. Числа располагаются в специально выделенном для них месте — регистрах арифметико-логического устройства (АЛУ). Только из регистров АЛУ черпает данные, а пересылку данных в регистры и изъятие результата выполняет устройство управления.

Во всяком случае именно так обстояло дело с первыми вычислительными машинами. Как мы уже знаем, элемент памяти машины способен хранить только два состояния, поэтому и разряд числа может принимать два значения — «0» и «1». Это заставило использовать для представления чисел в ЭВМ двоичную систему счисления. Она отличается от десятичной только количеством различных цифр, составляющих число: если в десятичной системе счисления,

а)

+		СЛАГАЕМОЕ									
		0	1	2	3	4	5	6	7	8	9
СЛАГАЕМОЕ	0	0	1	2	3	4	5	6	7	8	9
	1	1	2	3	4	5	6	7	8	9	0
	2	2	3	4	5	6	7	8	9	0	1
	3	3	4	5	6	7	8	9	0	1	2
	4	4	5	6	7	8	9	0	1	2	3
	5	5	6	7	8	9	0	1	2	3	4
	6	6	7	8	9	0	1	2	3	4	5
	7	7	8	9	0	1	2	3	4	5	6
	8	8	9	0	1	2	3	4	5	6	7
	9	9	0	1	2	3	4	5	6	7	8

б)

<div>+</div>		СЛАГАЕМОЕ	
		0	1
СЛАГАЕМОЕ	0	0	1
	1	1	0

*Правила сложения десятичных (а) и двоичных чисел (б)*

досчитав до девяти, мы записываем следующее число уже двумя разрядами (поскольку все различающиеся цифры уже исчерпаны), то в двоичной системе этот переход происходит уже на втором шаге счета. Таким образом, число 2 в десятичной системе будет иметь вид 10 в двоичной. В остальном эти системы счисления ничем не различаются, а таблицы основных операций значительно проще для двоичной системы счисления. Последнее суммирование дает два разряда, а значит, только при суммировании двух единиц будет перенос в старший разряд числа.

Производить вычитание в двоичной системе тоже проще, чем в десятичной, но вычитания большинство современных ЭВМ и вовсе не знает. Вместо вычитания выполняется сложение с отрицательным числом, что, как известно, одно и то же. Осталось разобраться в том, как же записывается отрицательное число. Для этой цели существует несколько способов. Рассмотрим один из них, наиболее распространенный. Различать положительное и отрицательное числа договорились по значению старшего разряда, который, как и любой другой разряд, может иметь два значения: либо «0», либо «1». Положительному числу соответствует ноль в старшем разряде, а отрицательному — единица. Абсолютное значение числа инвертируют, т. е. значение каждого разряда меняют на противоположное (вместо «1» пишут «0», вместо «0» — «1»), и добавляют единицу в младший разряд; таким образом получают представление отрицательного числа в ЭВМ.

Например, требуется сложить числа  $+3$  и  $-3$  в машине с четырехразрядным словом. Число  $+3$  будет представ-



а)

		ВЫЧИТАЕМОЕ									
		0	1	2	3	4	5	6	7	8	9
УМЕНЬШАЕМОЕ	0	0	9	8	7	6	5	4	3	2	1
	1	1	0	9	8	7	6	5	4	3	2
	2	2	1	0	9	8	7	6	5	4	3
	3	3	2	1	0	9	8	7	6	5	4
	4	4	3	2	1	0	9	8	7	6	5
	5	5	4	3	2	1	0	9	8	7	6
	6	6	5	4	3	2	1	0	9	8	7
	7	7	6	5	4	3	2	1	0	9	8
	8	8	7	6	5	4	3	2	1	0	9
	9	9	8	7	6	5	4	3	2	1	0

б)

		ВЫЧИТАЕМОЕ	
		0	1
УМЕНЬШАЕМОЕ	0	0	1
	1	1	0

Правила вычитания десятичных (а) и двоичных чисел (б)

лено в машине в виде 0011. Для отрицательного числа будут сделаны следующие преобразования:

- 11<sub>2</sub> — число —3 в двоичном коде,
- 0011<sub>2</sub> — абсолютное значение числа —3 в машине,
- 1100<sub>2</sub> — инверсия (обратный код) числа 3,
- 0001<sub>2</sub> — единица, добавляемая к обратному коду,
- 1101<sub>2</sub> — дополнительный код числа —3.

Складывая +3 и —3: 0011<sub>2</sub> + 11;1<sub>2</sub> = 10000. Поскольку левая единица выходит за пределы четырехразрядного машинного слова, то в ответе получим 0. Необходимость дополнительного кода для представления отрицательных



Сложение с отрицательным числом в ЭВМ

чисел вызвана тем, что при использовании обратного кода без прибавления единицы в младший разряд появились бы два значения для нуля:  $+0$  и  $-0$ , которые в машинном слове выглядели бы следующим образом:  $0000_2$  и  $1111_2$  соответственно.

В схемах умножения в основном используется особенность двоичной системы счисления. Для того чтобы умножить число на степень 2, достаточно сдвинуть число влево на значение показателя степени и справа дописать нужное количество нулей. Разумеется, это справедливо и для десятичной системы счисления, и Вы часто этим пользуетесь:

$$123 \cdot 10^4 = 1\,230\,000.$$

Но в отличие от десятичной системы в двоичной в каждом разряде может быть только 0 или 1, а значит, каждый разряд указывает на наличие или отсутствие соответствующей позиции степени 2. Таким образом, умножение чисел производится последовательным сдвигом влево и сложением результатов.

Пусть надо перемножить числа  $a$  и  $b$ :  $c = ab$ . Запишем числа в виде разложения по степеням двойки:

$$\begin{aligned} a &= a_n \cdot 2^n + a_{n-1} \cdot 2^{n-1} + \dots + a_1 \cdot 2 + a_0; \\ b &= b_n \cdot 2^n + b_{n-1} \cdot 2^{n-1} + \dots + b_1 \cdot 2 + b_0; \end{aligned}$$

Коэффициенты этих полиномов представляют собой значения двоичных разрядов чисел, а следовательно, равны 0 или 1.

Тогда результат можно представить в виде

$$c = ab_n \cdot 2^n + ab_{n-1} \cdot 2^{n-1} + \dots + ab_1 \cdot 2 + ab_0.$$

Отсюда легко понять суть операции: там, где  $b_i = 1$ , следует сдвинуть число влево на значение показателя соответствующей степени, а затем сложить результаты всех сдвигов.

Сравним два способа. При умножении через сложение число операций суммирования должно быть равно значению меньшего из сомножителей. При использовании операций сдвига можно уменьшить число сложений до количества ненулевых разрядов одного из сомножителей и затем выполнить столько же операций сдвига.

Операция деления доставила разработчикам ЭВМ еще больше забот, чем умножение. Для деления также использовали возможность сдвига на степень 2. Но при делении

$$10 \times 9 = ?$$

$$\begin{array}{r} + 1010 \\ 1010 \\ \hline ? \end{array}$$

а)

$$\begin{array}{r} + 1010 \\ 1010 \\ \hline 10100 \\ + \\ 1010 \\ \hline 11110 \\ + 1010 \\ \hline 101000 \\ + 1010 \\ \hline 110010 \\ + 1010 \\ \hline 111100 \\ + 1010 \\ \hline 1000110 \\ + 1010 \\ \hline 1010000 \\ + 1010 \\ \hline 1011010 \end{array} \quad \left. \vphantom{\begin{array}{r} + 1010 \\ 1010 \\ \hline 10100 \\ + \\ 1010 \\ \hline 11110 \\ + 1010 \\ \hline 101000 \\ + 1010 \\ \hline 110010 \\ + 1010 \\ \hline 111100 \\ + 1010 \\ \hline 1000110 \\ + 1010 \\ \hline 1010000 \\ + 1010 \\ \hline 1011010 \end{array}} \right\} 8 \text{ операций}$$

б)

$$\begin{array}{r} 1010 \\ + 1010 \\ 1010 \\ \hline 1011010 \end{array} \quad \left. \vphantom{\begin{array}{r} 1010 \\ + 1010 \\ 1010 \\ \hline 1011010 \end{array}} \right\} 3 \text{ операции}$$

$$= 90$$

Выполнение умножения: а — последовательным суммированием; б — сдвигами и одним суммированием

образуется остаток, и, кроме того, деление на нуль дает неопределенный результат.

Итак, мы уже набрали несколько элементарных операций, выполняемых арифметико-логическим устройством ЭВМ. Полный набор таких операций называют системой команд, а схемы их реализации составляют основу АЛУ. К базовым операциям относят сложение, вычитание, умножение, деление, операции сдвига чисел влево и вправо, сравнение, инверсию числа. Очень часто используются операции сложения с 1 (т. е. положительное приращение) и вычитание 1 (отрицательное приращение).

Помимо арифметического устройства АЛУ включает и логическое устройство, предназначенное для операций, при осуществлении которых отсутствует перенос из раз-

И			
		0	1
	0	0	0
	1	0	1

ИЛИ			
		0	1
	0	0	1
	1	1	1

Правила логического умножения И и логического сложения ИЛИ

ряда в разряд. Иногда эти операции называют логическое И и логическое ИЛИ.

Арифметико-логическое устройство в некоторых типах ЭВМ расширено по сравнению с обычными машинами. Существуют специальные устройства для выполнения матричных операций.

## 2.3. УПРАВЛЕНИЕ

Устройство управления занято выдачей команд для АЛУ, для чтения или записи чисел в память, для управления процессом ввода — вывода. Это же устройство выдает команды и для себя. Чем же руководствуется при этом устройство управления? Конечно же, программой, которая находится в памяти ЭВМ. Начинается процесс управления с того момента, когда получена первая исходная команда от человека «выполнить программу». Еще совсем недавно можно было найти такую ЭВМ, которая начинала выполнение любой программы после нажатия специальной кнопки «Пуск». Сегодня ЭВМ оснащают специальными программными средствами — операционными системами. Их задача вначале ограничивалась только запуском программы. Сейчас функции операционных систем расширились и специализировались настолько, что стоимость их разработки превышает стоимость собственно ЭВМ.

Итак, устройство управления выполняет программу, выбирая команды одну за другой из памяти. Но если бы этот процесс не был соответствующим образом усовершенствован, то программы имели бы необозримую длину и не умещались бы в памяти. Поскольку такой режим, конечно, неэкономичен, в программах содержатся команды, изменяющие последовательность выбора операторов программы из памяти. Менять следование операторов — одна из задач устройства управления, но предусматривает эти изменения человек при составлении программы. Существует несколько способов изменения порядка следования команд; цель их введения одна — выбор части программы, которая должна быть использована в данный момент времени. В частности, это могут быть те же операторы, которые только что были выполнены. Тогда мы имеем дело с циклом. Как при езде на велосипеде колеса оборот за оборотом крутятся вокруг своей оси и в то же время движутся вперед, так и в устройстве управления одни из операторов повторяют раз за разом одно и то же действие, а другие определяют движение вперед — тот момент, когда

надо будет двинуться согласно программе дальше. Достигается это тем, что в систему команд включается команда передачи управления или перехода к выполнению оператора с каким-то определенным номером по порядку.

Такой переход может совершаться либо всегда, либо по условию, которое вырабатывается при исполнении предыдущего оператора. Для некоторых ЭВМ переход по условию записывается в одну команду, для других — в две, тогда первая делает пропуск следующей, если выполнено условие, а следующая команда представляет собой просто безусловный переход.

Пример такого условного оператора дает нам команда наращивания на единицу и пропуска следующей, если в результате наращивания получили 0. Например, если какую-то часть программы надо выполнить 0357 раз, то сначала инверсируем это число и прибавляем единицу. Получили дополнительный код: 9642. После 357 прибавлений единицы получим число 9999, и следующий шаг даст 0 (разумеется, в машине все это делается в двоичном коде).

Еще одна команда, вырабатывающая условие, — это сравнение. Такая команда позволяет менять «направление» работы программы в зависимости от совпадения или несовпадения чисел, используемых в сравнении.

В памяти ЭВМ выделена специальная ячейка — регистр номера команды. Число, записанное в этом регистре, указывает номер команды в программе, которую необходимо выполнить. После выполнения команды номер в регистре увеличивается на 1. Микропрограмма команды безусловного перехода записывает другой номер в регистр и тем самым меняет ход исполнения программы. В случаях перехода по условию происходит либо увеличение этого регистра на 1, либо запись нового номера.

Еще одна важнейшая функция управляющего устройства, появившаяся у ЭВМ относительно недавно, — обработка прерываний. Вначале это была вынужденная мера для реагирования на необычные ситуации: деление на 0, получение несуществующего в машине адреса данного или команды и т. д. Но впоследствии прерывания стали использовать для обслуживания сразу нескольких пользователей сначала при обращении одного из них к медленным внешним устройствам, а затем и просто по очереди через короткие промежутки времени. Дело в том, что внешние устройства работают медленнее АЛУ, да и человек реагирует на изменение ситуации гораздо медленнее, чем вы-

числяет ЭВМ. В связи с этим, сохраняя внешне ощущаемую человеком непрерывность исполнения программы, машина может делить время выполнения, давая немного поработать по очереди всем программам.

Обработка прерываний в устройстве управления довольно проста: как мы, прервав на время чтение, кладем закладку в книгу, чтобы отметить нужную страницу, так и устройство управления запоминает место (регистр номера команды), на котором остановилась одна программа, и передает управление в операционную систему. Таким образом, устройство управления само не решает, кто будет следующим из пользователей машины,— эта сложная задача возложена на операционную систему. Своего рода «закладкой» для запоминания места приостановки выполнения программы служит специальный регистр — слово состояния, которое содержит основные параметры выполняемой программы.

Вообще существуют такие ЭВМ, которые имеют очень малую оперативную и внешнюю память, очень небольшой объем арифметических и логических команд, но зато обеспечены развитым устройством управления, специализированным для определенных целей. Примером могут служить станки с числовым программным управлением, которые в последнее время все шире применяются в народном хозяйстве.

Объединенные устройство управления и арифметикологическое устройство принято называть процессором. В отличие от ЭВМ расхождений в наименовании процессора на различных языках нет. Этот лингвистический факт свидетельствует прежде всего о четком понимании функций процессора, главная из которых заключается в обработке данных, хранящихся в памяти.

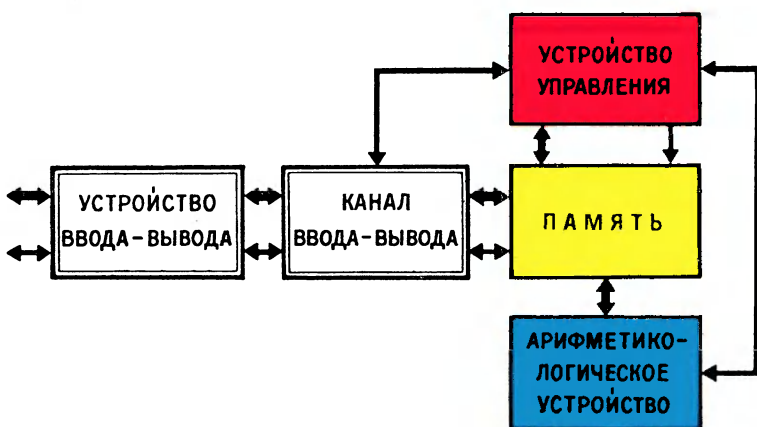
Кроме процессора в состав ЭВМ входят оперативная память, внешняя память и различные устройства ввода — вывода. Если оставить в стороне вопросы специализации процессора, то можно утверждать, что основное разнообразие применений ЭВМ связано с устройствами ввода — вывода (УВВ). В то же время вследствие этого разнообразия соединение УВВ с процессором стандартизировано, и потому различные ЭВМ могут иметь похожий набор УВВ. Это значит, что тип ЭВМ вполне определяется типом процессора. В одной ЭВМ можно использовать несколько процессоров. В частности, они могут быть и разнотипными, например один из процессоров — для управления всеми УВВ.

## 2.4. ВВОД — ВЫВОД

В вычислительных машинах с неймановской архитектурой ввод и вывод информации осуществлялся через арифметико-логическое устройство, поэтому на время ввода и вывода вычисления прекращались. Такая структура ЭВМ обладала существенными недостатками, один из которых заключался в снижении производительности во время ввода и вывода информации из-за необходимости подробного контроля этих операций и управления ими со стороны устройства управления.

В целях повышения производительности вычислительных машин в середине 50-х годов было осуществлено совмещение во времени процессов ввода — вывода и вычислений путем организации непосредственного обмена с памятью при вводе и выводе, минуя арифметико-логическое устройство. Основным шагом в этом направлении явилось включение в состав машины независимого канала ввода — вывода, представляющего собой фактически небольшую специализированную ЭВМ. Канал выполняет функции ввода — вывода одновременно (параллельно) с работой арифметико-логического устройства и обеспечивает независимый доступ к памяти и автономное управление операциями ввода — вывода. Вычислительная машина может быть оборудована несколькими каналами, способными осуществлять свои функции одновременно.

Итак, функция устройства ввода — вывода состоит в обеспечении информационной связи между человеком и



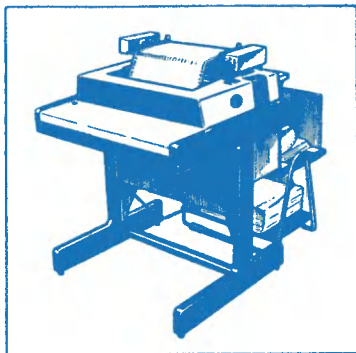
Структурная схема ЭВМ с каналом ввода — вывода

ЭВМ. Но информация может быть представлена в различных формах, например: в виде текста, чертежа, рисунка, графика, звукового сигнала или механических воздействий.

Почему ЭВМ оказывается полезной в самых разных областях, использующих различные формы представления знаний? Как это согласуется с всеобщей специализацией знаний? Один из важных

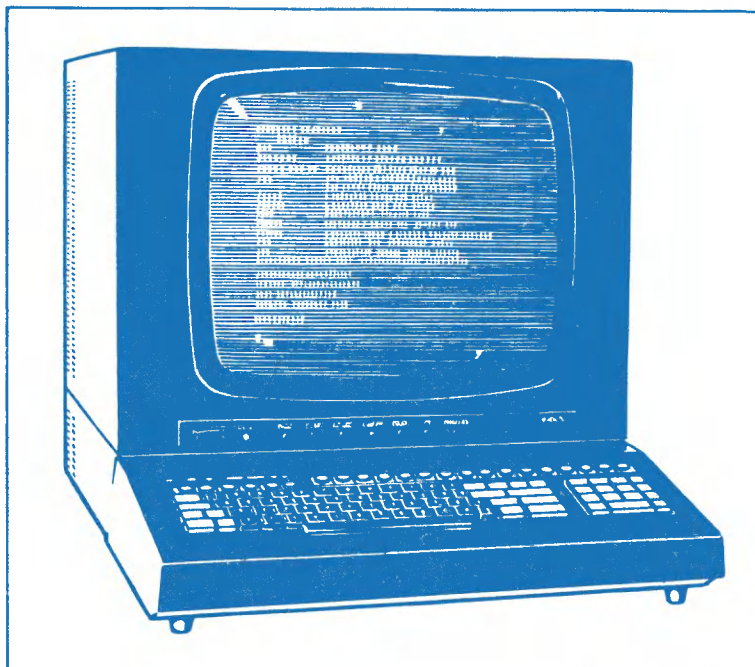
факторов «могущества» ЭВМ заключен в совместимости с ней самых разнообразных приборов, в какой-то мере имитирующих органы чувств человека, а в ряде случаев дополняющих их. Привычные нам источники информации — книги, рисунки, слуховые и тактильные ощущения — стали доступны и для «восприятия» ЭВМ через разнообразные устройства ввода. Сейчас можно работать с дисплеем (комбинация телевизора и клавиатуры пишущей машинки), с устройством графического ввода и графопостроителем, выводящим информацию в виде чертежа, с устройством речевого ввода и многими другими приборами.

А л ф а в и т н о - ц и ф р о в о й д и с п л е й — самый распространенный вид оконечного (терминального) устройства. Перед программистом находится клавиатура, подобная той, что используется в обычной пишущей машинке, но дополненная некоторыми специальными клавишами. Эти клавиши позволяют перемещать курсор — светящуюся метку — в горизонтальном и вертикальном направлениях. Набранный текст высвечивается на экране. Если программист заметил ошибку в набранном тексте, он подводит курсор к ошибочной позиции и исправляет написанное. После того как программист убедился, что набранный текст правилен, он нажимает клавишу и тем самым отправляет свою директиву на исполнение ЭВМ. Реакция машины будет выдана на экран дисплея. Часто дисплейный терминал снабжают устройством печати. Тогда программист имеет возможность всю или часть своей работы в диалоговом режиме распечатать на бумаге — получить твердую копию, как говорят программисты. Отлаженную и надежно работающую программу можно записать в библиотеку и вызывать по мере необходимости через



*Печатающее устройство*

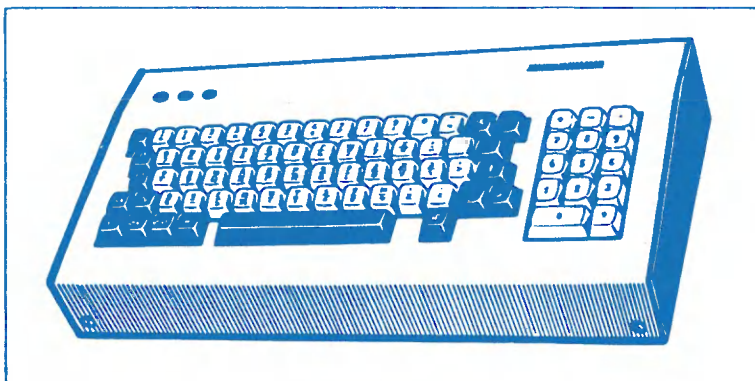




*Алфавитно-цифровой дисплей*

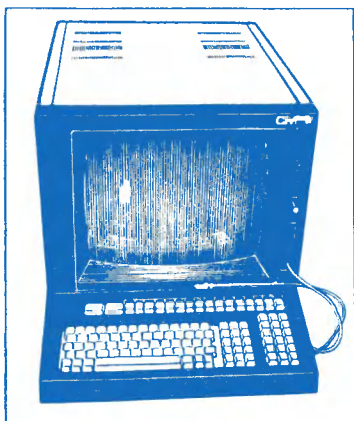
терминал. Алфавитно-цифровой дисплей очень удобен для редактирования программ, работы в диалоговом режиме с системами обработки информации, с базами данных.

Если данные, с которыми работает программист, содержат в основном графическую информацию, то



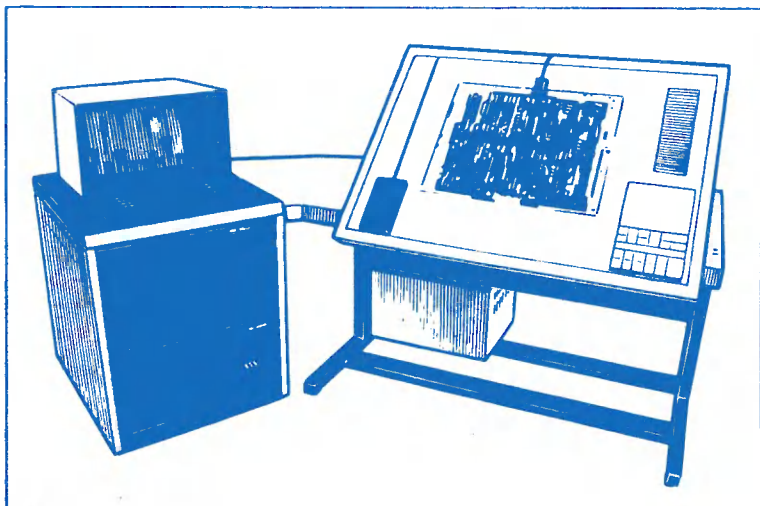
*Клавиатура дисплея*

применяют графические дисплеи, в которых предусмотрена возможность «рисования» картинок. Процесс рисования можно организовать по-разному. Часто уже имеются базовые элементарные конфигурации, на основе которых формируется сложный рисунок. Тогда на дисплей можно вызывать эти элементы и надстраивать их к рисунку в соответствующем месте и масштабе. Если программа создает изображение, то она его также выдает на экран. Этот тип терминала используют в проектировании, картографии, при обработке изображений.



*Графический дисплей*

Графический терминал содержит дисплей и (или) графопостроитель. Из набора базовых элементов, при желании пополняемых, можно строить на экране любые рисунки и чертежи. Такой терминал необходим технологу, конструктору, дизайнеру, архитектору и еще многим специалистам. Получив желаемое изображение, человек дает команду изготовить ее копию на бумаге. Если терми-



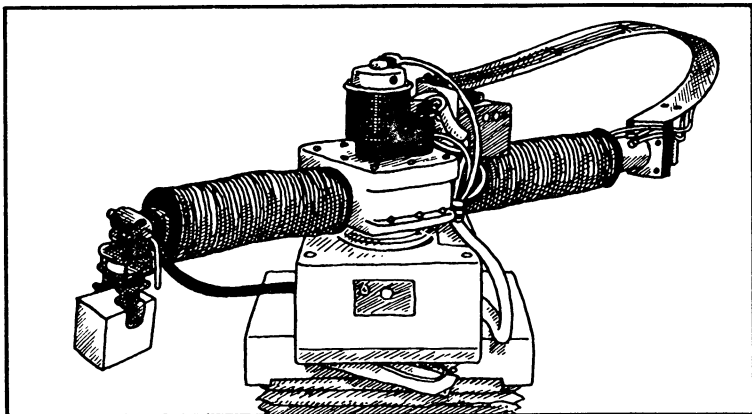
*Графический терминал*

нал подключен к ЭВМ, то вместе с «картинкой» машина выдает и рассчитанные параметры новой конструкции. Работая в диалоговом режиме, человек может перебрать значительно больше вариантов и не только визуально, но и математически точно определить их качество.

Информация бывает представлена в таком виде, что ее текстовое или изобразительное представление становится довольно трудоемким (например, температурные данные или характеристики твердости материала). В этом случае гораздо удобнее, минуя дисплей, передавать информацию от соответствующего датчика в цифровом виде непосредственно в ЭВМ. Там информация будет обработана, а результат представлен в виде либо карты, либо текста и графиков. Возможность вводить информацию в ЭВМ от различных датчиков широко используется в космических и глубоководных исследованиях, производственных процессах, робототехнике.

Мы знаем, что в ЭВМ можно ввести информацию и произвести ее обработку в доли секунды. Очень важно, чтобы ЭВМ не только выполняла свое основное назначение (поиск информации, моделирование ситуаций, вычисление), но и выдавала результат в той форме, которая облегчала бы ее восприятие человеком и при которой не требовались бы длительные расшифровки, соизмеримые по времени с самим вычислением. Внешнее устройство должно быть ориентировано на общение с человеком. Такое общение может происходить в различных формах. Очень объемная численная информация может быть представлена в виде графиков или гистограмм. Состояние модели, имитирующей процесс в реальном времени, можно передать цветом или определенными контурами так, чтобы человек моментально уловил угрожающую ситуацию. Результат проектирования вовсе не обязательно выдавать в виде текста и чисел. Гораздо удобнее, если машина сама построит чертеж на графопостроителе и предоставит необходимые спецификации комплектующих деталей.

В некоторых областях использования вычислительной техники на устройство ввода — вывода возлагается столь большая нагрузка, что устройство ввода — вывода стали рассматривать как самостоятельное, а все остальные блоки ЭВМ — как вспомогательные. Такими устройствами являются терминалы, т. е. оконечные устройства, работающие с ЭВМ, манипуляторы и роботы. Роботы-«интеллектуалы» автоматически выполняют операции, связанные с деятельностью, считавшейся всегда прерогативой человека. К ним с полным основанием можно отнести автомати-



*Робот-манипулятор*

ческие системы чтения чертежей и проектирования, системы оценки сложных процессов, автоматические библиографы и др.

Разнообразие устройств ввода — вывода по всей видимости в дальнейшем будет быстро возрастать. Ведь их нужно столько, сколько форм представления знаний используют люди и сколько существует типов специализированных технологических процессов.

## **2.5. ВЫЧИСЛЯТЬ, ПОМНИТЬ И УПРАВЛЯТЬ**

Способность вычислять, помнить, управлять последовательностью событий — в чем здесь сходство и различие между человеком и машиной? Почему машине вообще доступны столь сложные действия?

Видимо, все дело в особой организации основных блоков, постоянно обращающихся друг к другу, на которые возложены функции: помнить то, что вычисляется и каким образом следует производить вычисления; управлять порядком вычислений и вычислять место в памяти; искать целенаправленно и притом быстро. Управление — не какая-то заранее и раз и навсегда заданная последовательность операций. Управление осуществляется за счет памяти, на основе уже законченных действий с возможностью вычислять изменения будущей последовательности действий. Именно взаимосвязь всех действий при выполнении машиной задачи позволяет проводить определенные аналогии между функциональными способностями чело-

века и ЭВМ. Такого глубокого взаимодействия частей нет, пожалуй, ни в каком другом техническом устройстве. Как ни странно может показаться на первый взгляд, именно вследствие глубины взаимодействия составляющие ЭВМ элементы очень просты.

Рассмотрим взаимосвязь о существовании трех видов памяти: оперативной, внешней и долговременной. Любой из этих видов памяти имеет собственное устройство управления, которое координирует выполнение основных команд взаимодействия с памятью (чтение и запись), и некоторые дополнительные, обеспечивающие, с одной стороны, соответствующие скоростные характеристики основных операций, а с другой, работоспособность самого устройства внешней памяти. Кроме того, в устройство управления внешней памятью входит блок вычислений места на носителе информации по указанному численному адресу (физический адрес).

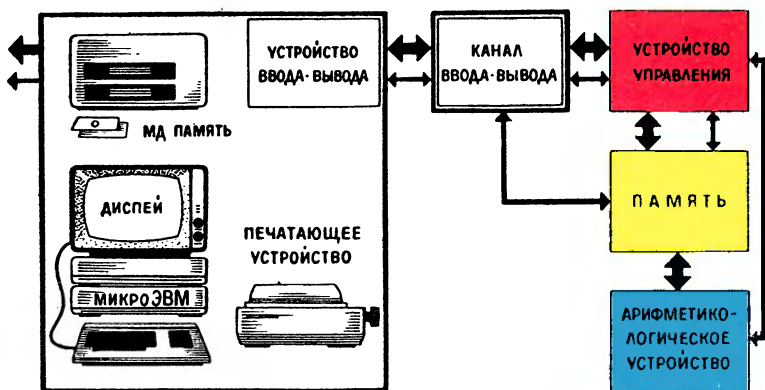
В свою очередь устройство управления ЭВМ, как мы уже отмечали, имеет собственную внутреннюю память: регистры для счета по программе, счета номеров команд, хранения слова состояния и адресов операндов и др. Кроме набора регистров в устройстве управления предусмотрены блоки для простейших вычислений внутреннего пользования (например, счетчики).

И наконец, арифметико-логическое устройство помимо схем выполнения команд имеет некоторую собственную память для осуществления промежуточных операций вычисления. В последнее время набор команд в ЭВМ (система команд) выполняется микропрограммами, так что команда ЭВМ, посылаемая в микропрограммный блок, представляет собой просто имя микропрограммы. Таким образом, внутри АЛУ как бы находится специализированная вычислительная машина, программами которой являются системы команд основной ЭВМ.

Аналогичное взаимопроникновение блоков есть и в устройствах ввода — вывода, для управления которыми все чаще в состав ЭВМ включают специализированный процессор ввода — вывода.

Итак, расчленив ЭВМ на основные блоки, в каждом из них мы снова обнаруживаем ЭВМ с полным комплектом блоков!

Рассмотренные нами блоки имеют две составляющие: аппаратную и программную. Нагрузка на эти составляющие в различных ЭВМ распределяется по-разному. Чем более гибким, управляемым должно быть функционирование того или иного блока либо ЭВМ в целом, тем более развитой



*Расчленив ЭВМ на основные блоки, в каждом из них снова обнаруживаем ЭВМ с полным комплектом блоков*

будет программная составляющая. Если же ЭВМ предназначена для решения узкого круга задач, то те общие операции, которые одинаково производятся в каждой отдельной задаче, могут быть жестко фиксированы в самой аппаратной конструкции ЭВМ. Необходимость в этом продиктована тем, что аппаратная часть значительно дешевле программной, и тенденция к снижению ее стоимости продолжает сохраняться.

## 2.6. МИКРОПРОЦЕССОРЫ

Микропроцессор базируется на логических схемах того же типа, что и центральный процессор ЭВМ. В обоих случаях для манипулирования данными и выполнения вычислений под управлением программы используются цифровые схемы. Однако их размеры существенно различаются. Микропроцессор, как правило, выполняется на одной или двух микросхемах. Почти все микропроцессоры изготавливают на кремниевых кристаллах. Длина стороны куба-кристалла равна 0,64 см. Такой кремниевый кристалл с содержащимися в нем электрическими схемами микропроцессора упаковывается в корпус так называемой интегральной схемы, имеющей от 16 до 64 выводов.

Под мощностью микропроцессора понимают его способность обрабатывать данные. Ее принято оценивать тремя основными характеристиками: длиной слова данных, количеством адресуемых слов памяти и скоростью выполнения команд.

Наиболее часто микропроцессоры сравнивают по длине слов данных. Первоначально были разработаны 4-разрядные микропроцессоры, которые и сейчас еще находят применение. Четыре бита — это длина двоично кодированного десятичного числа. В калькуляторах, промышленных системах управления и некоторых других устройствах микропроцессор оперирует только с числами. В таких случаях использование 4-разрядного микропроцессора является идеальным решением. Широкое распространение получили 8-разрядные микропроцессоры, стоимость которых невысока. Их достоинства заключаются в следующем: во-первых, 8-битовое слово позволяет «упаковать» в нем два 4-битовых слова, и, во-вторых, 8 бит позволяют представить любой из используемых символов языка (букву, цифру, знак). Большинство первых 16-разрядных микропроцессоров выпускали как элементы мини-ЭВМ в виде больших интегральных схем.

Другим мерилom мощности микропроцессора служит число слов или байтов памяти, к которым он может адресоваться. Каждому слову в памяти присваивается номер его местоположения — адрес. Чем больше значение максимального адреса памяти, тем больше вычислительная мощность микропроцессора. Для 4-разрядных микропроцессоров обычным является диапазон адресации от 4 до 8 К слов памяти. Для 8-разрядных микропроцессоров верхняя граница этого диапазона составляет 65 К слов. Что же касается 16-разрядных микропроцессоров, то их диапазон адресации неизмеримо шире: от 32 К до 4 М слов \*.

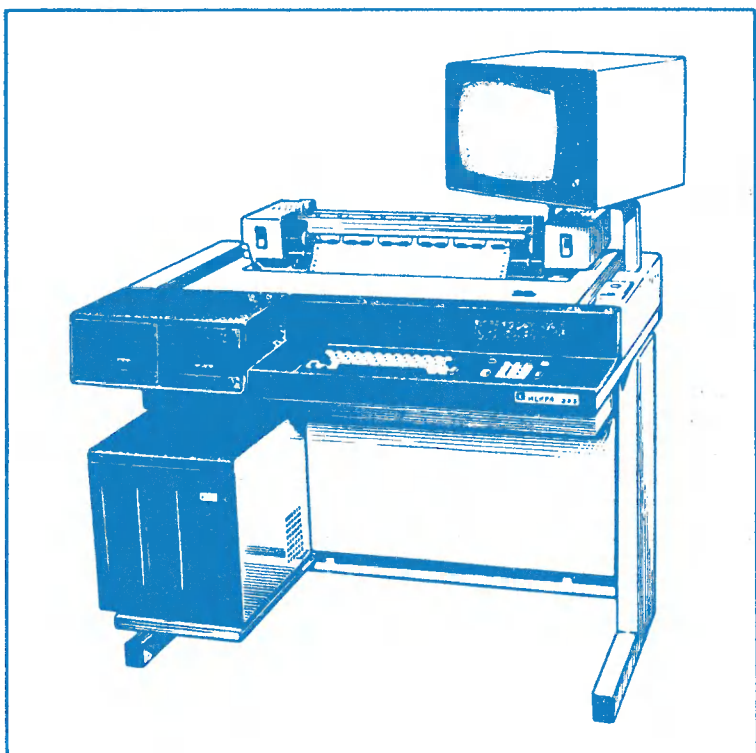
Третьим параметром, характеризующим мощность микропроцессора, является скорость, с которой он выполняет команды. Она оценивается продолжительностью цикла «выборка — выполнение» для одного шага программы.

Рассмотренные величины служат для сравнения мощности и обычных процессоров.

Часто термины «микропроцессор» и «микроЭВМ» используют как синонимы, однако они имеют различное смысловое значение. Микропроцессор — это интегральная схема, предназначенная для обработки данных и управления. Что же касается микроЭВМ, то она представляет собой законченную вычислительную систему, центральной частью которой является микропроцессор. В системе, кроме того, входят оперативная память, постоянная

---

\* Кслов — килослов = 1000 слов, Мслов — мегаслов =  $10^6$  слов.



*МикроЭВМ «Искра-555»*

память (ПЗУ), устройства ввода — вывода, соединяющие микроЭВМ с терминалом. Вычислительные системы организованы так, чтобы пользователь не ощущал разделения средств на программные и аппаратные. Обычно у программистов нет повода для выяснения точной границы аппаратного и программного обеспечения. Более того, бурно развивающееся микропрограммирование размывает некогда существовавшую четкость функционального различия этих средств.

Массовое производство и применение поставило новые сложные задачи перед пользователями микропроцессорной техники. Ведь вокруг больших ЭВМ собраны высококвалифицированные специалисты, а при внедрении микропроцессора трудно рассчитывать на профессиональную поддержку и программистскую подготовку работающих с этой техникой. По этой причине микропрограммирование делится на два направления. Во-первых, использо-



вание сложного и обширного программного обеспечения большой ЭВМ и, во-вторых, активная работа в диалоговой системе программирования (Бэйсик) на самой микроЭВМ.

Объединяя сотни и тысячи микропроцессоров между собой, удается создавать вычислительные средства для задач, решение которых недоступно даже для очень больших однопроцессорных ЭВМ. Успех здесь связан прежде всего с тем, что такие системы проектируются специально под вполне определенные алгоритмы для узких классов задач. Кроме того, на таких микропроцессорных системах достигают огромного быстродействия прежде всего за счет глубокого распараллеливания вычислений.

Сегодня микропрограммная техника стала основой практически всех блоков ЭВМ. Процессор, в котором микропрограммы интерпретируют систему команд, терминальные комплексы, устройства ввода — вывода, канал, управление внешней памятью — все эти средства в той или иной мере берет на вооружение микропрограммирование. В целом программирование на всех уровнях от внешнего языка до системы команд предоставляет необычайно широкие возможности в области специализации ЭВМ.

# 3

## ЯЗЫК, ЧЕЛОВЕК, ЭВМ

### 3.1. ПУТЬ ЗАДАЧИ К ЭВМ

Желание достигнуть взаимопонимания с машиной заставило человека размышлять над тем, что он считал очевидным, ординарным. Сейчас такой, казалось бы, естественный для человека процесс, как взаимопонимание, активно изучается под разным углом зрения — начиная с лингвистических аспектов и кончая проблемами, связанными с созданием искусственного интеллекта. Сложность этого вопроса обусловлена существенным различием в степени формализации языков (инструментов общения) человека и ЭВМ.

В машине все конструкции языка жестко заданы системой команд и способом работы процессора. А на естественном языке одну и ту же информацию можно передать по-разному. Если возникает некоторая неясность, мы находим в языке средства уточнить, пояснить непонятную мысль, по возможности сформулировав ее другим образом. Но даже при переводе фразы с одного естественного языка на другой неизбежно что-то теряется. В. Я. Брюсов, человек, энциклопедически образованный, знавший два десятка языков, в статье «Фиалки в тигеле» привел следующее высказывание знаменитого английского поэта Перси Шелли, образно отражающее трудности стихотворного перевода: «Стремиться передать создание поэта с одного языка на другой,— это то же самое, как если бы бросили в тигель фиалку, с целью открыть основной принцип ее красок и запаха. Растение должно возникнуть вновь из собственного семени или оно не даст цветка,— в этом-то и заключается тяжесть проклятия вавилонского смешения языков» \*.

---

\* П. Шелли имеет в виду библейский миф о попытке построить Вавилонскую башню до небес. Разгневанный дерзостью людей бог «смешал» их языки, и они перестали понимать друг друга.— *Прим. ред.*



#### Путь задачи к ЭВМ

При использовании ЭВМ речь идет не только о разных языках, но и о разных уровнях формализации. Перевод здесь сопровождается потерями. Чтобы таких потерь было меньше, перевод осуществляется в несколько этапов.

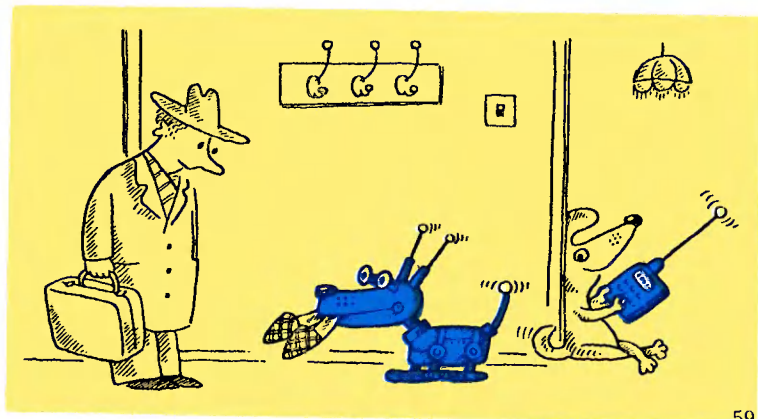
Специалисты по разработке средств общения ЭВМ вынуждены были пойти на компромисс и создать так называемые языки высокого уровня. По внешнему виду они напоминают естественный язык, а потому человеку гораздо легче читать и писать тексты на таком языке, чем в кодах машины. Все предложения строятся по четко определенным правилам, не допускающим многозначности; синтаксически предложения жестко фиксированы. Их кажущаяся «естественность» — всего лишь замаскированная схематичность. Однако все же эти языки отстоят несколько дальше от машинных языков. Например, в них уже не обязательно указывать адреса ячеек, в которых хранится информация. Следовательно, в этом смысле они становятся «машинонезависимыми». Языки высокого уровня призваны отразить логическую структуру данных и действий над ними, не затрагивая структуру их хранения в ЭВМ. Но чтобы ЭВМ получила указание о том, как разместить данные и какие из них передавать на обработку, подключается набор программ, обеспечивающий трансляцию и загрузку программы в ЭВМ, а значит,

оперируя с текстом на языке высокого уровня, машина берет работу по трансляции на себя.

Информация при передаче ее ЭВМ претерпевает ряд изменений. Некоторая исходная идея формулируется на языке, принятом в данной области деятельности. Затем эта форма представления информации переводится на язык высокого уровня и вводится в ЭВМ. И наконец, сама ЭВМ переводит представление информации с языка высокого уровня в машинный язык, оперирующий понятиями «адрес ячейки», «присвоение», «операции арифметики и логики» и некоторыми другими. Таким образом, задача на пути ввода ее в ЭВМ проходит несколько этапов формулирования, каждый из которых имеет собственный язык.

### 3.2. ЯЗЫК И ИНФОРМАЦИЯ

В процессе постепенного освоения способов работы с ЭВМ раскрылись такие их способности, о которых поначалу и не подозревали. Действительно, современные ЭВМ могут практически все, но только при одном условии — если сам человек четко осознает, что он хочет, а главное, умеет объяснить машине, каким образом можно это сделать. А чтобы объяснить что-либо машине, необходимо записать в нее информацию. Информация — это любой дифференцирующий признак, который может приводить к изменениям. В качестве дифференцирующего признака выступает знак как символ информации, а возможность ее изменений предполагает необходимость модификации знаков. Одна и та



же запись информации может иметь различные смысловые значения.

Слово «информация» сейчас стало очень широко употребляемым. Часто говорят и об информационном взрыве (т. е. огромном потоке информации), и об информационном голоде. Казалось бы, это взаимоисключающие понятия! Как можно испытывать информационный голод при изобилии информации? Оказывается, можно. И основная причина этого кроется в способах представления и упорядочения информации.

Для нас основной способ хранения и передачи информации — язык устный или письменный. Информацию об одной и той же вещи можно представить по-разному даже на одном и том же языке. В то же время весьма отдаленные идеи могут иметь сходное внешнее представление. «Мысль изреченная есть ложь» — эти слова Ф. И. Тютчева отражают сложность однозначного выражения и восприятия идеи, мысли, информации в языковой форме. Иногда свойство неоднозначности нашего языка расценивают как полезное, поскольку оно определяет гибкость, адаптируемость и, главное, развитие языка, иногда — как крайне нежелательное, поскольку затрудняет восприятие и осмысление конкретных понятий. Хотя каждый человек может в той или иной степени добиться успеха в формулировании своих идей на нашем естественном языке, все же неточность в восприятии — это объективная закономерность.

Каким же образом знания передаются от одного человека к другому и будущим поколениям? Исторически сложились два основных способа передачи информации. Один из них — путь логических выводов — всегда требует ограниченного числа четко сформулированных исходных положений и набора объектов, к которым применимы эти положения. На этот путь раньше всех встала математика. Формулировка и доказательство теоремы — это своеобразный текст на языке математики. Но хотя логическое (аксиоматическое) построение текстов и приводит к однозначному их пониманию, оно не очень удобно для восприятия человеком: слишком много определений и правил необходимо помнить. Этот путь в основном используют, когда необходимо небольшое количество исходных положений. Подавляющее же большинство своих навыков действовать адекватно различным ситуациям мы приобретаем из текстов на естественном языке, например из художественной литературы, а не из формальных инструкций.

В противоположность логическому способу построения высказываний лингвистический способ не предполагает однозначного определения составляющих элементов. Многие слова имеют большой спектр значений и их оттенков. А когда слова составляют фразу, то из всего спектра значений выделяется единственно приемлемое в данном контексте, т. е. смысл предложения выявляется пересечением смысловых оттенков составляющих слов, словосочетаний. В полном объеме однозначно осмыслить текст, построенный лингвистическим способом, не представляется реальным. Например; в художественной литературе автор ставит своего героя в самые разнообразные ситуации, сталкивает с различными персонажами, благодаря чему у нас складывается определенный его образ. То, что читатели зачастую «спорят» о произведении, как раз и свидетельствует об отсутствии единого восприятия всех смысловых оттенков.

Но то, что хорошо в литературе, явилось бы крахом в математике или любой другой формальной системе. Весьма плачевной была бы ситуация, если бы ЭВМ «траговала», например, инструкции процессора каждый раз по-другому.

В отличие от ЭВМ человек обладает способностью гибко перестраивать способ выражения мысли. И все многообразие привычных нам средств выражения мысли мы хотели бы сделать «понятным» для ЭВМ. Возможно ли это в полном объеме? Не утратим ли мы исходный смысл при переводе привычных нам форм представления знаний на язык, понятный ЭВМ? Чтобы ответить на эти вопросы, рассмотрим самые характерные черты машинного и естественного языков.

### 3.3. ЯЗЫК, ПОНЯТНЫЙ ЭВМ

Машинный язык вследствие своей однозначности восприятия и четкости формулировок, несомненно, относится к аксиоматическому типу. Собственно, текст на машинном языке — это набор двоичных чисел, т. е. сочетания «0» и «1». Кодирование информации в двоичный алфавит особых затруднений не вызывает. Отсутствуют и потери информации. Программисты только в крайнем случае распечатывают тексты программ на машинном языке, или, как они говорят, «в кодах». Такое отношение к кодам машины вызвано необходимостью монотонного и скрупулезного перехода к удобной человеку символической содержательной форме. Представим все же, что мы решили получить распечатку. На листе бу-

000021	010400	020000	121073	040461	020040	020040	000144
000000	000000	000003	000143	000000	000000	000000	000000
000000	000000	000021	000007	003400	040001	140562	040461
020040	020040	000000	000007	000052	025000	100015	133670
027106	046520	020014	027104	046104	020003	027104	051524
020004	027130	051525	041006	027104	043105	051002	027104
044517	027007	027130	044517	027010	027122	044517	027011
027104	052101	027012	027123	052117	050015	043114	047501
052013	041514	051111	047401	051440	020040	020005	000052
000073	035400	060150	163563	000000	010302	000000	016001
000003	016002	000070	030302	000075	016003	000100	016004
000073	133022	062000	000103	072000	000102	016005	000020
000065	030222	000073	016006	000105	000070	000065	120132
062000	000105	003000	002021	026000	000031	026000	000063
121022	062000	000110	006400	016007	000112	000043	103030
016010	000065	016011	000073	016012	131030	062000	000111
016013	016014	000073	016004	000073	000031	014400	060115
136102	000050	030232	000073	016002	000070	000065	062000
000102	132120	042000	000103	072000	000102	003004	042000
000104	002021	121000	026000	000014	002400	016015	000031
000044	022000	060132	165612	000112	000000	024065	054054
021120	044475	021054	000000	043061	033056	030164	026065
054054	000000	021160	047440	070154	047575	040544	000000
064442	026106	030461	027061	026042	000000	026565	063517
066170	044151	045501	000000	021051	000044	000012	005000
060105	120210	000075	000000	000000	000000	000000	040000
000006	000012	000007	003400	060102	060236	000103	000000
000001	000030	000007	000007	003400	060102	060222	000110
000000	000006	000002	000007	000004	002000	120001	120001
000000	000004	000000	000000	000021	010400	020000	132130
051440	020040	020040	000216	000000	000000	000007	000143
000000	000000	000000	000000	000000	000000	000021	000007

### *Программа в машинных кодах*

маги столбцы чисел, цифры и только цифры, никаких знаков пунктуации, ничего похожего на предложения пусть даже непонятного, иностранного языка, нет примечаний и комментариев, все строки имеют одинаковую длину. Скудная картина!

Однако текст имеет вполне определенные форму и содержание, есть и начало, и конец, и разделители. Содержание (чаще говорят — семантика) программы состоит в предписании порядка и способа работы процессора. Что касается формы, то она только на первый взгляд примитивна. Можно не только выделить самостоятельные фрагменты в рассматриваемой числовой последовательности, выражающие завершенное действие (вычисление) ЭВМ, но и провести аналогию между текстом программ на машинном языке и текстом на естественном языке с позиций их структуры. Отдельные части машинной программы, т. е.

блоки (или, точнее, процессы), будут выполнять роль предложений языка ЭВМ, поскольку они задают определенные действия процессора.

Говоря о смысле, семантике языковых конструкций, подразумевают процесс восприятия и интерпретации текста, в котором одинаково важны и источник текста, и его приемник. В ходе взаимодействия человека с ЭВМ интерпретация команд программы процессором позволяет выявить и реализовать семантику, действия, запрограммированные человеком. На машинном языке не так уж много основных конструкций, задающих элементарные действия процессора. Богатые возможности предоставляют их комбинации. Конечно, если есть предложения (процессы), то должны быть и другие составляющие текстов, например разделители, заменяющие точку и запятую. Их роль на уровне машинного языка выполняют команды передачи управления и остановки процессора.

Итак, основные структурные единицы языка ЭВМ — это процессы, т. е. последовательность команд из основного набора и команды передачи управления и остановки. В современных машинах команда «Стоп» выполняет двойную роль: для одних процессов это реальная остановка работы, но в большинстве случаев ее назначение сводится к передаче управления операционной системе. Здесь имеется тонкое различие: команда «Стоп» всегда завершает работу программы, в то время как передача управления завершает один процесс и начинает другой.

Фундаментальное значение в вычислительной технике имеет процесс, окончание которого сопровождается передачей управления первой команде этого же процесса. Процессор цикл за циклом повторяет все команды от первой до последней вплоть до того момента, когда значение какой-либо переменной не удовлетворит заданному условию, и процесс завершается. Сам процесс называют циклическим, а соответствующую переменную — переменной цикла.

Именно циклические процессы за счет быстродействия ЭВМ позволяют решать сложные задачи, многократно выполняя простые, элементарные действия и операции. Цикл — многократное исполнение одних и тех же команд — напоминает, несомненно, основной рабочий цикл процессора, в течение которого обрабатывается любая команда из системы команд ЭВМ. Однако по отношению к процессору это внешняя программа.



Роль циклического процесса в обработке информации с помощью ЭВМ сравнима со значением, которое имело изобретение колеса в механике, да и в целом в истории техники. Колесо, вращательное движение, цикл Карно, циклы в компьютере — все эти понятия одной природы. Циклы и периодичность — явления, широко распространенные в органическом и неорганическом мире. В программировании цикл также можно отнести к самому распространенному типу языковых конструкций машинного языка. Не следует путать повторяемость работы процессора с повторяемостью самого текста. Иначе говоря, цикл в большей степени отражает семантическую сторону кодов ЭВМ, нежели синтаксическую особенность языка машины. Поясним сказанное, обратившись к нашему естественному языку. В предложениях «они играли в мяч, пока не стемнело», «за окном вагона до самого Курска мелькали распаханное поля», «три месяца, каждую вторую среду он ходил смотреть фильм „Девушка моей мечты“, но на связь с ним никто так и не вышел» легко отделить слова, которые служат обоснованием повторяемости и прекращения повторения одних и тех же действий. При этом в самих формулировках повторы отсутствуют. Среди слов машинного языка, входящих в описание циклического процесса, столь же легко выделить те команды и переменные, которые необходимы только для определения момента прекращения цикла.

Обратите внимание на третье предложение: если бы встреча со связным состоялась, то не пришлось бы идти на все условленные сеансы. Вы уже поняли, что речь идет о раннем завершении циклического процесса.

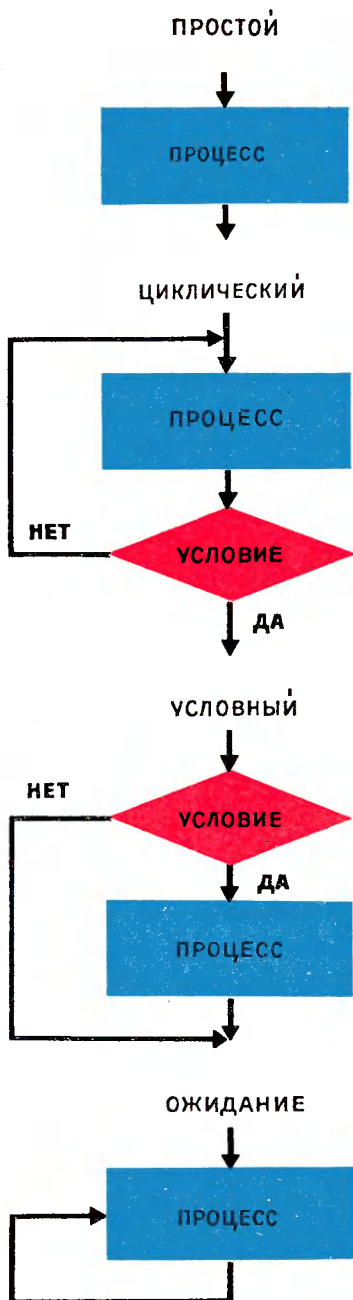
Еще один тип предложения машинного языка — условный процесс, т. е. процесс, который начинается условной передачей управления последней команде. Заметим, что структурно условный процесс является полной инверсией циклического процесса. Первый начинается какой-либо командой и завершается передачей управления в начало, второй начинается с передачи управления в конец. Главное в условном процессе, однако, то, что он предваряется анализом некоторого условия. В нашем языке наиболее близкий эквивалент — предложение со структурой «Если ..., то ..., иначе ...».

Итак, основное назначение такого типа процесса заключено в возможности его выполнения только при определенном заранее условии. И если цикл — движитель вычислений, то условный процесс — это руль управления вычислениями. К характерному признаку циклического

процесса следует отнести передачу управления назад, т. е. к командам, уже выполнявшимся, а к особенностям условного процесса — передачу управления вперед, т. е. пропуск определенных команд.

В отличие от циклического и условного процессов последовательность команд без передачи управления будем называть линейным процессом. Он обычно отделен в программе другими процессами. В пределе, когда в программе вообще нет передач управления, вся программа представляет собой один линейный процесс.

Пожалуй, перечень возможных ситуаций будет полным, если дополнить его передачей управления на самого себя. Такая на первый взгляд нелепая конструкция языка ЭВМ не только существует, но и довольно часто используется в любой современной операционной системе с обработкой прерываний. Ее называют динамическим остановом или ожиданием. Дело в том, что процессор может не иметь для выполнения ни одной команды из задач пользователей и самой операционной системы; тогда необходимо ждать. После того как произошло какое-либо



внешнее относительно ЭВМ событие или закончился некоторый отрезок времени, ожидание прерывается и начинается счет — выполнение процессором команд.

В итоге имеем четыре базовых процесса: линейный, циклический, условный, ожидание.

Если процессы напоминают предложения естественного языка, то составляющие их элементы аналогичны словам. Они так и называются — машинные слова. Машинное слово содержит либо команды и адреса других машинных слов, в которых находятся величины, предназначенные для вычислений, либо конкретные числа, называемые константами.

В подобном разделении слов программы также прослеживается аналогия с естественным языком, в котором одни части речи (глаголы) выражают способ действия, а другие (существительные) описывают объект действия.

Но если цели машинного и естественного языков во многом схожи, поскольку отражают общее назначение языка — передавать информацию, то способы достижения этих целей существенно различаются. И хотя каждый человек способен овладеть естественным языком, еще раскрыты далеко не все принципы его образования и развития.

#### 3.4. ЯЗЫК ЕСТЕСТВЕННЫЙ И ИСКУССТВЕННЫЙ

В интересную «лингвистическую» ситуацию ставит современный американский писатель-фантаст Роберт Шекли героя своего рассказа «Потолкуем малость». До прибытия на планету На Фред Джексон — опытейший специалист по установлению контактов с инопланетянами — даже не предполагал о возможности существования такого языка, которым он не смог бы овладеть в кратчайший срок. Ведь «у него были поразительные способности к языкам и сверхъестественная интуиция на значения слов. Когда Джексон сталкивался с непонятным языком, он быстро и безошибочно вычленял значащие единицы — основные «кирпичики» языка. В предложении он с легкостью выделял информационную часть, случаи модального употребления и эмоциональную окраску. Его опытное ухо сразу же различало грамматические явления».

Правда, и на планете На язык (хон) ее обитателей сначала показался Джексону несложным. Через несколько дней он овладел им настолько хорошо, что достаточно

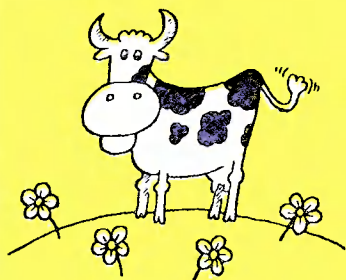
свободно мог объясняться с наянцами. Однако вскоре Джексон с досадой и удивлением обнаружил, что число непонятных ему в языке хон слов стало лавинообразно возрастать и смысл речи начал ускользать от него. Казалось, чем больше он учил язык, тем меньше его знал. Это было столь невероятным, что Джексон заподозрил наянцев в искусственном усложнении языка с целью уйти от контактов с посланцем другой планеты. «Но поверьте мне, я ни в коей мере не собираюсь *нумнискатерить!* И не *нонискаккекаки*, и Вы действительно должны этому *дебрушили*», — убеждал Фреда Джексона обескураженный его гневными упреками собеседник-наянец.

Джексон решил не сдаваться и, убежденный в своих способностях в конце концов осилить этот странный, не подчиняющийся, с его точки зрения, никаким разумным правилам язык, снова засел за работу. Каково же было удивление Джексона, когда он обнаружил, что за несколько дней его добровольного затворничества язык хон изменился до неузнаваемости!

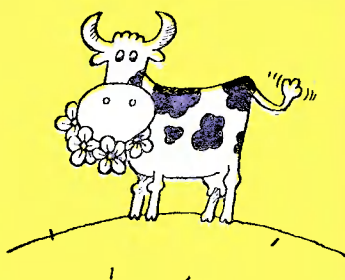
«Сейчас этот язык состоял из единственного слова „ман“. Оно могло иметь самые различные значения, в зависимости от высоты тона и порядка слов, от их количества, от ударения, типа повтора и ритма, а также от сопровождающих жестов и выражения лица. Язык, состоящий из бесконечных вариаций одного и того же слова! Джексон не хотел верить этому, но он был слишком хорошим лингвистом, чтобы сомневаться в том, о чем говорили ему собственные чувства и опыт... То, что случилось, было в некотором смысле неизбежным: ведь изменяются все языки. Но на Земле и на нескольких десятках миров, с которыми она установила контакты, этот процесс был относительно медленным. На планете На это происходило быстрее. Намного быстрее. Язык хон менялся, как на Земле меняются моды, только еще быстрее. Он был так же изменчив, как ... погода... Он менял свои формы, как меняет свои очертания снежная лавина. Рядом с ним английский язык казался неподвижным ледником...

Наблюдатель вроде Джексона не мог даже надеяться зафиксировать или выделить хотя бы одно звено из динамично движущейся цепи терминов, составляющих этот язык... Сам факт подобных изменений делал недоступным как наблюдение за языком, так и выявление его закономерностей. Все попытки овладеть языком планеты На разбивались об его неопределимость».

Джексон почувствовал себя побежденным и посрамленным. Не сказав больше ни слова, он повернулся и



Я увидел ее на поляне  
с цветами



Я увидел ее на поляне  
с цветамии

пошел к своей ракете. «Через полчаса корабль стартовал, а еще через пятнадцать минут лег на курс».

Итак, героя рассказа Р. Шекли сначала смутила подвижность языка. Но в совершенно безвыходное положение он попал, столкнувшись с полисемией, т. е. с неоднозначностью смысловых оттенков одного и того же слова в зависимости от контекста.

Сложна для восприятия и структурная неоднозначность, когда от логического группирования разных слов при прочтении зависит интерпретация предложения, например: «Я увидел ее на поляне с цветами»\*.

Еще труднее уловить неоднозначность смысловую, которая возможна в ряде случаев, несмотря на отсутствие структурной неоднозначности. Например, из предложения «Цыплята готовы к обеду» нельзя сделать однозначный вывод о судьбе цыплят.

Четвертый тип неоднозначности — семантическая неоднозначность. Скажем, слово «ели» может быть существительным множественного числа или глаголом в прошедшем времени.

Еще один тип неоднозначности — прагматическая неоднозначность — определяет трудность в соотнесении местоимения со словом, к которому оно относится. Так, в предложении «он уронил ящик на стол и сломал его» однозначно неясно, сломан ящик или стол.

\* Пример этот заимствован из литературы и является классическим, хотя в первом из двух проиллюстрированных на рисунке случаев правильнее было бы сказать «Я увидел ее на цветущей поляне» или «Я увидел ее на поляне в цвету». — *Прим. ред.*

Но многозначность — не единственный источник сложности формального перевода. Правила грамматики служат основой анализа предложений, но их недостаточно для синтеза. Даже при анализе мы привлекаем много «интуитивных» понятий, никак не отраженных в грамматике. Например, почему в словосочетаниях «мама пришла» и «папа пришел» окончания глаголов разные? Ответ прост — су-

ществительные имеют разный род. Это понимаем мы, но не ЭВМ. По виду слов ЭВМ не различила бы их род. Даже если в ЭВМ будут храниться грамматические категории всех слов, то и тогда не удастся избежать бессмысленных фраз, в то же время грамматически безупречных. Например, «жук поймал мальчика» вместо «мальчик поймал жука».

Фраза, абсолютно корректная синтаксически, может быть лишенной смысла. Впрочем, и программа, корректная с позиций языка программирования, может не соответствовать никакому алгоритму. ЭВМ способна облегчить деятельность человека по проверке корректности программ только на уровне грамматики, но не смысла.

Язык, понятный ЭВМ, однозначен. Все его правила строго определены и согласованы между собой. «Словарь» языка машины (т. е. система команд) чрезвычайно скуден, обычно он не превышает 500 слов. Словарь же естественного языка составляет сотни тысяч слов.

Таким образом, вследствие смысловых, структурных и словарных различий машинного и естественного языков необходим многоэтапный перевод формулировки задачи на естественном языке в машинное представление.



### 3.5. ПОСТАНОВКА ЗАДАЧИ ДЛЯ ЭВМ

Человек анализирует некоторую ситуацию и выделяет из нее конкретный ограниченный набор сведений, фактов, известных закономерностей, а в итоге формулирует задачу на естественном языке.

Такого рода формулировки можно найти и в учебнике математики: «Из пункта *A* в пункт *B*...», «В бассейн вливается из одной трубы...», «Вычислить число  $\pi$ ...».

Число  $\pi$  связывает радиус и площадь круга в формуле  $S_{кр} = \pi R^2$ . Если взять круг единичного радиуса ( $R=1$ ), то  $S_{кр} = \pi$ . Площадь круга можно рассматривать как предел, к которому стремится площадь вписанного  $N$ -угольника при  $N \rightarrow \infty$ . Возьмем последовательность вписанных в единичный круг  $N$ -угольников при  $N = 4, 8, 16, 32, \dots$  Тогда площадь очередного  $N$ -угольника приближенно можно рассматривать как площадь круга ( $S_N \approx S_{кр}$ ) и использовать для вычисления приближенного значения числа  $\pi$ .

Площадь  $N$ -угольника вычислим по формуле

$$S_N = \frac{1}{2}[(X_N - X_1)(Y_N + Y_1) + \sum_{i=2}^N (X_{i-1} - X_i)(Y_{i-1} + Y_i)].$$

Примем следующие упрощения:

1) будем вычислять только четверть площади  $S_N$ , попадающую в первый квадрант системы координат; тогда  $S_N = 4S_K$ , где  $K = N/4 + 2$ ;

2) координаты вершин  $K$ -угольника определим так:

$X_1=0, Y_1=0$  — центр окружности;

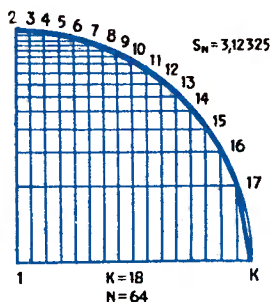
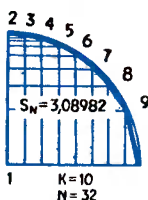
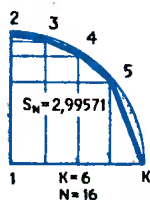
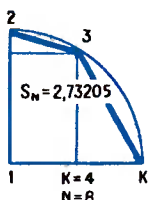
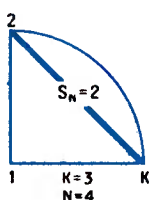
$X_2=0, Y_2=1$  — верхняя точка окружности;

$X_K=1, Y_K=0$  — крайняя левая точка окружности;

значение  $X$  промежуточной точки между 2-й и  $K$ -й получаем равномерным делением радиуса на  $K-2$  частей, тогда

$$X_i = (i-2)/(K-2), \quad Y_i = \sqrt{1 - X_i^2} = \sqrt{1 - \left(\frac{i-2}{K-2}\right)^2} = \frac{\sqrt{(K-2)^2 - (i-2)^2}}{K-2} = \frac{\sqrt{(K+i-4)(K-i)}}{K-2}, \quad \text{т. е.}$$

$$X_i = \frac{i-2}{K-2}, \quad Y_i = \frac{\sqrt{(K+i-4)(K-i)}}{K-2}.$$



Первые пять шагов построения и вычисления площади  $K$ -угольника

```

0001  FTN4
0002  С ПРОГРАММА ВЫЧИСЛЯЕТ
0003  С РЯД ПРИБЛИЖЕНИЙ К ЧИСЛУ ПИ.
0004      PROGRAM A1
0005      DOUBLE PRECISION PI,SI,S
0006      REAL N
0007      SI=0D0
0008      N=4.0
0009      DO 1 I=1,24
0010          PI=S(N)
0011      IF(PI,LE,SI) GOTO 2
0012          WRITE(6,3)PI,N
0013          N=2*N
0014          SI=PI
0015 1      CONTINUE
0016 2      STOP
0017 3      FORMAT(5X,"PI=",F16,14,5X,
0018          *"ПО ПЛОЩАДИ",F11,1,"-УГОЛЬНИКА")
0019      END

```

*Программа A1 вычисляет ряд приближений числа  $\pi$*

Учитывая методы или алгоритмы решения данной задачи и исходя из ее содержания, выбирают подходящий формальный язык.

Формальных языков очень много, почти столько, сколько существует областей деятельности человека. Они возникают и развиваются по мере расширения и уточнения наших знаний.

Для этих языков характерна детерминированность, т. е. однозначное соответствие текста и смысла. В связи с этим уже возможен переход к языку программирования, который по сути тоже относится к формальным языкам. Наряду с декларативным описанием задачи он содержит также императивную часть, которая указывает непосредственно порядок действий для достижения цели (решения).

Применительно к задаче вычисления  $\pi$  подходит язык программирования ФОРТРАН, специально разработанный для записи математических формул. Операторы, помеченные буквой «С», служат комментарием к программе. Функция  $S(N)$  вычисляет площадь  $N$ -угольника.  $S(N)$  получает значения координат  $K$ -угольника при обращении к процедуре  $\text{COORD}(X, Y, I, K)$ .  $\text{COORD}$  — имя процедуры;  $X, Y, I, K$  — формальные параметры процедуры. Функция  $S(N)$  подсчитывает площадь  $N$ -угольника как четыре площади  $K$ -угольника.

В таблице представлены все используемые в программе типы операторов с пояснением выполняемых ими действий. Справа в таблице приведены номера строк программы, в которых используется данный тип оператора.

Записи на языке программирования уже достаточно, чтобы машина решила поставленную перед ней конкрет-



# ОПЕРАТОРЫ, ИСПОЛЬЗОВАННЫЕ В ФОРТРАН-ПРОГРАММЕ

Структура оператора	Назначение оператора	Строка
<b>PROGRAM</b> <имя> *	<b>Описание</b> дает имя программе	4
<b>DOUBLE PRECISION</b> <список имен>	<b>Описание</b> задает тип переменной и функций с двойной точностью для перечисленных в списке имен	5
<b>REAL</b> <список имен>	<b>Описание</b> задает тип переменной «действительное число»	6
<имя> = <выражение>	<b>Оператор присвоения</b> переменной <имя> результата вычисления выражения в правой части. (В строке 7 переменной присвоен «0» с двойной точностью: $\emptyset D \emptyset$ .)	7, 8, 10, 13
<b>DO</b> <метка> <переменная цикла> = <начальное значение> , <конечное значение> <операторы цикла> ... <метка> CONTINUE	<b>Оператор цикла</b> задает условие повторения последовательности действий вплоть до метки, указанной после DO. При каждом исполнении цикла переменная цикла увеличивает свое значение на единицу	9 : 15
<b>IF</b> ( <имя> .GT. <имя> ) <оператор>	<b>Условный оператор</b> задает условие выполнения оператора, стоящего за скобками (GT — больше чем — от англ. greater then)	11
<b>GO TO</b> <метка>	<b>Оператор перехода</b> на строку с указанной меткой	11

WRITE (<номер> <метка> ) <список имен>	Оператор печати печатает значения переменных из списка имен; <номер> задает номер устройства, откуда берут данные для печати; <метка> указывает на строку программы, где задан формат печати	12
FORMAT <список форматов>	<p>Описание формата представления вводимых или выводимых на печать значений. В кавычки заключаются части текста, печатающиеся без изменений. В строке 17 задан следующий формат:</p> <p> <math display="block">PI=0.\underbrace{0000000000000000}_{F16.14} \underbrace{0000000000000000}_{\text{по площади}} \underbrace{0000000000000000}_{F11.1} \underbrace{0000000000000000}_{\text{"-УГОЛЬНИКА"}}</math> <math display="block">5X \text{ "PI=" } F16.14 \quad 5X \text{ "по площади" } F11.1 \text{ "-УГОЛЬНИКА"}</math> </p> <p>5X — пять пробелов; «PI=» — три символа PI=; F16.14 — 16 позиций для действительного числа с десятичной точкой, из них 14 занимает дробная часть; 5X — пять пробелов; «по площади» — 10 символов текста; F11.1 — одиннадцать позиций действительного числа, из которых одну занимает дробная часть; «-угольница» — 10 символов текста</p>	17, 18
STOP	Конец исполнения программы	16
END	Конец текста программы	19
* Для представления структуры оператора используют скобки <...> с описанием части оператора, а в программе на их месте пишут конкретные имена.		

ную задачу. Но перед ее выполнением машина сделает ряд преобразований текста: синтаксический разбор, генерацию объектного кода, оптимизацию кодов.

Конечным результатом этих преобразований всегда будет программа на языке машины, т. е. в кодах ЭВМ.

### 3.6. ТРАНСЛЯЦИЯ И ИСПОЛНЕНИЕ

Среди всех программ машины выделяются специальные — трансляторы или целые комплексы программ — диалоговые системы. В их задачу входит преобразование текстов, построенных человеком и воспринимаемых машиной. Иначе говоря, с помощью трансляторов и диалоговых систем ЭВМ выполняет рутинную работу по перекодированию, замене фраз внешнего языка общения на процессы, кодовые последовательности машинного языка. Программное обеспечение диалога человека с машиной может образовать несколько слоев, в каждом из которых участвует свой транслятор или другая подобная программа.

Машина же всегда выполняет программу только в кодах, которые получаются в результате трансляции программы, записанной на внешнем языке программирования. Хотя на этапе трансляции ЭВМ берет на себя утомительную работу по кодировке описанных пользователем действий в машинные коды, это не означает, что человеку на этапе нечего делать. Как только что-либо будет непонятно ЭВМ, она тут же сообщит об ошибке и будет беспомощно ждать следующих указаний.

Фортран-транслятор обнаружил в программе две ошибки: неправильное имя оператора и неправильную запись условия. Сообщение об ошибке выдается сразу после ее обнаружения. В заключение транслятор сообщает общее количество ошибок в программе.

Дело в том, что в машине хранятся таблица всех возможных для данного языка высокого уровня команд и соответствующие им машинные коды. Ничего сверх того ЭВМ «не понимает». Если пользователь ошибся в написании команды, то в таблице просто не будет соответствующей строки, а в этом случае осуществляется переход на строку с сообщением об ошибке. В таблице в написании предложений на языке программирования используется ограниченный набор имен — служебных (ключевых) слов. Например, в операторе  $IF(X.GT.Y)X=Y$  слово IF (если) означает, что далее в круглых скобках заключено условие

```

0001  FTN4
0002  С ПРОГРАММА ВЫЧИСЛЯЕТ
0003  С РЯД ПРИБЛИЖЕНИЯ К ЧИСЛУ ПИ,
0004  PROGRAM A1
0005  DOUBLE PRECISION PI,SI,S
0006  REEL N
0007  SI=0D0

      REEL N
      REEL?
**A1 ** ERROR 10 DETECTED AT COLUMN 10

0008      N=4,0
0009      DO 1 I=1,24
0010          PI=S(N)
0011      IF(PI < SI) GOTO 2

      IF(PI < SI) GOTO 2
      IF(PI <?
**A1 ** ERROR 53 DETECTED AT COLUMN 13

0012      WRITE(6,3)PI,N
0013      N=2*N
0014      SI=PI
0015  1    CONTINUE
0016  2    STOP
0017  3    FORMAT(5X,"PI=",F16,14,5X,
0018      * "ПО ПЛОЩАДИ",F11,1,"-УГОЛЬНИКА")
0019      END

```

*Программа A1 с синтаксическими ошибками*

исполнения следующего за закрывающей скобкой оператора, а слово GT (больше чем), ограниченное с двух сторон точками, отделяет два сравниваемых по величине операнда. Такой оператор соответствует привычной в математике записи: «Если  $X > Y$ , то  $X$  принимает значение  $Y$ ».

В качестве служебных слов языков программирования служат различные символы, например =, +, /, или слова естественного языка. Главное — точно определить место такого слова в структуре оператора и не ошибаться в его написании. В универсальных языках программирования стремятся использовать один и тот же набор служебных слов в различных версиях трансляторов, разработанных в разных странах, чтобы обеспечить совместимость программных продуктов. По этой причине сложилась традиция использовать в языках программирования англоязычные слова. (Это не означает, что готовый программный продукт в виде, например, диалоговой системы будет оперировать ими на внешнем уровне — они скрыты от пользователя, ими пользуется программист.) При разработке языков для узкого класса задач, не опи-

рающихся на накопленный набор программ, прибегают к национальным языковым средствам для упрощения восприятия текстов программ. (Отметим, что тем самым не удастся приблизить в желаемой степени машинный язык к естественному, поскольку набор слов очень строго ограничен, а ошибок при программировании зачастую наблюдается больше, чем при обращении к непривычным словам с четко заданным порядком их следования.)

Транслятор сопоставляет операторы в таблице со строками текста программы. Сейчас лишь часть синтаксических ошибок ЭВМ «умеет» исправлять за счет избыточности языков высокого уровня. На этом этапе процесс «объяснения» задачи машине завершается и можно переходить к решению задачи.

В чем же, собственно, заключается перевод текста с языка высокого уровня? Основная идея состоит в процессе подстановки вместо отдельных слов и целых предложений заранее подготовленных последовательностей команд (машинных слов). Решение вопроса, какой последовательностью кодов заменить ту или иную конструкцию, принимается разработчиком транслятора. Кроме разбора текста и подстановок, которые называют генерацией объектного кода, современные трансляторы производят оптимизацию полученной машинной программы.

В целом в процессе трансляции обычно выделяют следующие этапы:

1) синтаксический разбор, т. е. распознавание правильных синтаксических конструкций и сообщение об ошибках;

PI=2,0000000000000000	по площади	4,0=угольника
PI=2,73205080757157	по площади	8,0=угольника
PI=2,99570906810535	по площади	16,0=угольника
PI=3,08981914437013	по площади	32,0=угольника
PI=3,12325303784732	по площади	64,0=угольника
PI=3,13510242288601	по площади	128,0=угольника
PI=3,13929691278446	по площади	256,0=угольника
PI=3,14078079241867	по площади	512,0=угольника
PI=3,14130558296148	по площади	1024,0=угольника
PI=3,14149115278246	по площади	2048,0=угольника
PI=3,14155676659709	по площади	4096,0=угольника
PI=3,14157996547129	по площади	8192,0=угольника
PI=3,14158816768744	по площади	16384,0=угольника
PI=3,14159106790598	по площади	32768,0=угольника
PI=3,14159209347400	по площади	65536,0=угольника
PI=3,14159245687895	по площади	131072,0=угольника
PI=3,14159258571436	по площади	262144,0=угольника
PI=3,14159263402700	по площади	524288,0=угольника

*Результат вычислений по программе A1*

- 2) распределение памяти исходя из имен переменных, их описаний;
- 3) генерация объектного кода, т. е. выполнение подстановок;
- 4) оптимизация программы в объектном виде.

Вас, вероятно, озадачил термин «объектный вид». Это понятие относится к сфере действий операционной системы ЭВМ, для которой программа и есть объект работы. Трансляторы 50-х годов производили программу, которая могла быть тут же исполнена. В наше время это не так. Сегодняшний транслятор закончит работу, а программа только частично будет готова к исполнению.

Сейчас накопилось уже довольно много программ для часто используемых задач, поэтому деятельность программиста уже не сводится к скрупулезному написанию программы оператор за оператором. Работа программиста напоминает, скорее, действия опытного конструктора, создающего изделие из готовых узлов. Компоновку текста программы человек передает ЭВМ, указывая имена заимствованных программ и место их подсоединения. Собственно связывание частей в единую программу машина производит в кодах. Объединение программных текстов в машинных кодах позволяет компилировать результаты трансляции с разных языков высокого уровня. После завершения процесса компиляции программа готова к загрузке и исполнению.

Помимо компиляции в программировании используют режим интерпретации. Интерпретация отличается от компиляции тем, что транслятор подготавливает к исполнению каждый оператор программы отдельно по мере их поступления.

Такое программирование обеспечивает ряд преимуществ. Человек может исправлять ошибки, как только они обнаружатся. Он имеет возможность проверять работоспособность записанных операторов, не заканчивая всей программы. Диалоговое программирование легче в обучении, и программисты быстрее достигают поставленных целей.

Однако эффективные программы для многократного использования и крупные программные проекты делают не в диалоге с интерпретирующими системами, а пользуясь компиляцией.

Таким образом, подготовительный этап в большинстве случаев достаточно длителен и существенно превосходит по времени собственно решение. Вот почему ныне внимание все в большей степени акцентируется на проблеме: как

добиться того, чтобы быстро и достаточно полно переводить научные, технические и даже бытовые задачи на язык коротких и четких команд для ЭВМ. В достижении этой цели решающее слово принадлежит программистам. Программисты учат машину «мыслить». От мастерства программиста зависит, будет ли ЭВМ помощником человека или останется дорогостоящей, функционально недогруженной «игрушкой».

### 3.7. КАКОЙ ЯЗЫК ЛУЧШЕ?

Итак, каждому этапу формализации представления знаний в ЭВМ соответствует свой язык. Мы используем последовательно естественный язык, язык предметной области (в нашем примере — математики), язык высокого уровня и, наконец, машинный язык. Поскольку на ЭВМ решаются задачи, относящиеся к различным сферам науки и техники, то в целях экономии и удобства языки высокого уровня специализировали по предметным областям. Так, Фортран используется в основном для решения математических и физических задач. Существуют экономические языки программирования (например, Кобол), языки для моделирования (в частности, Снобол).

В зависимости от того, как представлены знания, в предметной области строятся и соответствующие языки программирования. Между процедурными и декларативными знаниями точно такая же разница, как между понятиями «вычислить» и «найти». Они равноможны по представимости и подчас незаметно переходят одно в другое. Сокращения размеров перебора достигают путем расширения объемов вычислений, и наоборот, длительные расчеты сокращаются введением различного рода таблиц, параметров и прочих описаний. Собственно, эффективность программирования в основном и зависит от точно угаданного соотношения объема вычислений и данных. Хотя все же надо признать, что декларации «натуральнее», они ближе к реальному миру: табличное задание функций первично по отношению к формульной записи.

Постепенно для каждой области стали создавать свои языки, и разработчик языка получил вполне конкретный «образ» той области, на которую рассчитан язык, с одной стороны, и системы машинных команд, с другой. При проектировании языка учитываются все возможные виды запросов со стороны предметной области. Кроме того, немаловажное значение имеет удобство языка. Чем ближе по

конструкциям язык высокого уровня к языку предметной области, тем меньше вероятность ошибок при написании программ и тем удобнее работать специалисту предметной области, который может и не знать всех тонкостей программирования и функционирования ЭВМ.

Сейчас при проектировании языков ЭВМ основное значение уделяется такому их свойству, как естественность восприятия и использования. Причем естественность отнюдь не означает стремления формализовать естественный язык. Скорее, речь идет о формализации языка предметной области, который сам по себе уже обладает гораздо в большей степени однозначностью и фиксированной терминологией, нежели естественный язык в полном объеме.

В свое время говорили, что «в каждой науке столько науки, сколько в ней математики», имея в виду строгость, точность знаний. Относительно применения ЭВМ к конкретной предметной области и проектирования предметно-ориентированного языка по аналогии можно сказать, что ЭВМ подготовлена к использованию в конкретной области ровно настолько, насколько формализованы в ней знания. Вот почему создание автоматизированных систем перевода текстов с одного естественного языка на другой оказалось сложной задачей, несмотря на то что ЭВМ может хранить большой объем различных словарей и быстро осуществлять поиск при переводе слов с одного языка на другой. С помощью языка мы формируем и накапливаем знания об окружающем нас мире, и именно знания причинно-следственных связей в той или иной предметной области помогают нам адекватно воспринимать тексты переводов с одного языка на другой. Сочетание знаков, символов и совокупность слов только тогда становятся текстом, когда несут читателю знание.

Даже универсализм естественных языков всегда порождал узкоспециализированные языки предметных областей — истории, химии, физики, математики и др. Естественное желание человека — тратить как можно меньше сил на изложение своей мысли и быть понятым (принцип наименьших усилий).



## 4

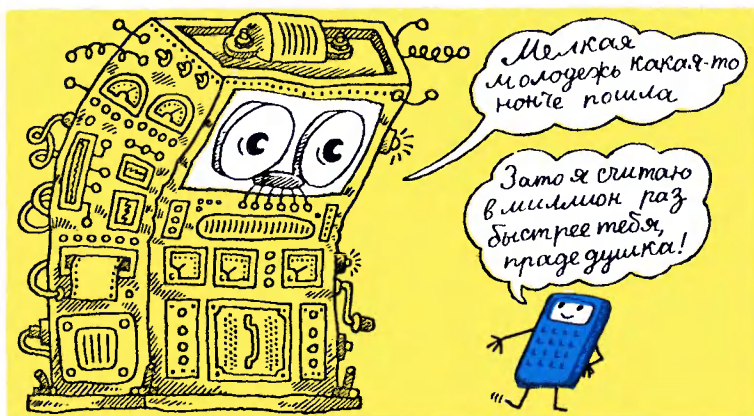
# ОТ ПОСТАНОВКИ ЗАДАЧИ К РЕШЕНИЮ ЕЕ НА ЭВМ

### 4.1. ЗАЧЕМ НУЖНЫ МОДЕЛИ?

Построение моделей — отличительная черта мышления человека. Благодаря постоянному расширению класса моделей — от простых копий и описания объективных законов природы до абстрактных представлений непосредственно не наблюдаемых процессов — развиваются наши способности постигать все более сложные явления, процессы окружающего мира. С помощью моделей мы не только познаем новое. Модели лежат в основе системы обучения и распространения знаний. Особенно ярко это проявляется в детстве. Игрушки — копии реального мира — играют важную роль в развитии и воспитании ребенка, выступая средством психологической подготовки его к будущим жизненным ситуациям. Сущность любой модели сводится к тому, чтобы выделить, воспроизвести только те свойства и характеристики реальных объектов, которые необходимы и достаточны для решения поставленной задачи.

Если модель адекватна поставленной задаче, то это служит залогом правильного, быстрого и легкого получения соответствующего результата. Часто говорят, что правильная постановка задачи — это уже половина решения, а остальное зависит от правильного выбора модели. Например, при решении в школе задач типа «Из одной трубы в бассейн вливается...» для вычислений не используют физическую модель, в которой учитывались бы все гидродинамические свойства жидкости и вихревых потоков.

Моделирование как один из методов научного познания задолго предварило появление ЭВМ. Известны различные типы моделей, например физические, математические. Сейчас активно развиваются так называемые информационные модели, в которых не воспроизводятся сами свойства и характеристики объекта моделирования, а



дается их описание на каком-либо языке. Точность и успешность моделирования в этом случае определяются только точностью описания объекта, его характеристик и связей между ними.

Построение модели всегда подразумевало необходимость строгой формализации анализируемого процесса. Именно формализация нашего эмпирического и интуитивного представления о некотором явлении, процессе позволяет нам предвидеть (прогнозировать) его поведение, тем самым проверяя и уточняя границы применимости данной модели.

Возникновение календарей — результат отражения модели чередования природных процессов — смены дня и ночи, приливов и отливов и т. д. Ритуальные танцы, игры — модели коллективного поведения в жизни общества.

Роль ЭВМ для моделирования трудно переоценить. Особое значение имеет возможность варьировать и уточнять исходные описания, облегчающие выявление существенных свойств исследуемого процесса. ЭВМ не только позволяет увеличить количество рассматриваемых параметров, но и допускает постепенное усложнение, уточнение описания по мере расширения знаний об объекте.

Наиболее быстрое признание моделирование на ЭВМ получило в тех областях, где уже имелись достаточно формализованные описания объектов: в механике, физике, технике. И это легко объяснимо, так как описание задач приводилось к числовой форме и требовался только кропотливый труд по проведению большого объема вычислений. С переходом от логарифмической линейки к ЭВМ изменились только количественные характеристики ско-

рости счета, а не качественные. Поэтому в этой части школьной программы вполне достаточно овладения программируемым микрокалькулятором.

ЭВМ же — даже сегодняшнего дня — это многофункциональный инструмент, который в последнее время распространяется и в таких областях знаний, как химия, биология, медицина, экономика, библиографический поиск, создание больших баз данных и т. д. Эти области знаний все еще математически слабо формализованы, и качество решения, как правило, ограничено не возможностями машинных расчетов, а несовершенством моделей. Если модель описывает явление неточно, то применение ЭВМ может лишь ускорить получение результата — как правильного, так и неправильного.

Идеи моделирования проникают и в гуманитарные области: литературу, историю, искусство, что, однако, также сдерживается неудовлетворительным уровнем формализации знаний в этих областях. Для устранения этой трудности есть два пути: либо строго формализовать задачи гуманитарных исследований, либо «подучить» ЭВМ и подготовить ее к «пониманию» нестрого сформулированных задач. Исследования велись в обоих направлениях. Пожалуй, сейчас уже ясно, что единственно возможным является путь «взаимных уступок». Действительно, любая задача, вводимая в ЭВМ, должна однозначно (в смысле исходной постановки вопроса) пониматься человеком. Другое дело, что ответ может не сводиться к количественному результату, а иметь форму текста, изображения, нескольких альтернативных решений. Исходя из выданной машиной информации можно снова поставить вопрос, и, таким образом, в результате «взаимного обучения» ЭВМ поймет, что она должна делать, а человек воспримет результат, который позволит принять решение о следующих шагах.

Задачу математического моделирования не могут решить одни специалисты по вычислительной технике и математики — для составления модели необходим большой объем знаний о самом объекте исследований. Моделирование является примером сложной межотраслевой деятельности. Здесь необходимо объединить усилия специалистов предметной области (физиков, математиков, химиков) и программистов.

Интеллектуальным ядром моделирования является триада «модель — алгоритм — программа», без такого триединства невозможен успех в решении сложных задач на ЭВМ. Сотрудничество на базе использования ЭВМ спе-



циалистов конкретной области знаний и специалистов по моделированию порождает определенные трудности, в связи с чем требуется их многосторонняя подготовка: и научная, и организационная, и даже психологическая.

Первый шаг на этом пути — изучение информатики в школе. Школьники должны быть готовы к систематическому изучению методов работы с ЭВМ в соответствии с триадой «модель — алгоритм — программа». Как показывает опыт, для этого недостаточно ознакомиться с основами программирования. Более того, вредно сводить проблеме использования вычислительной техники к проблеме обучения программированию. Нужно формировать структурно-логическую способность мышления, что позволит строить удачные модели изучаемых явлений и исследовать их с помощью вычислительной техники.

Математическая модель начинает формироваться с того момента, когда формулируется система аксиом, описывающая не только сам объект, но и некоторую совокупность правил, определяющих допустимые операции над этим объектом. По мере накопления фактов модель постепенно развивается в математическую теорию, которая сама начинает служить новым источником информации, подсказывает существование новых феноменов, требует опытного изучения тех или других процессов, определяет возможность создания и развития новой техники. Именно формализованные модели оказываются одним из важнейших инструментов технического прогресса. ЭВМ, в свою очередь, тоже является инструментом, с одной стороны, порожденным техническим прогрессом, а с другой, рез-

ко интенсифицирующим его рост. В этой положительной обратной связи и состоит до конца пока не осознанный феномен влияния ЭВМ на все стороны жизни человеческого общества.

Мы еще раз хотим обратить внимание читателя на то, что специфическая особенность функционирования ЭВМ посредством программирования, т. е. подробного инструктивного — шаг за шагом — представления всех этапов ее работы, вызвала необходимость использования алгоритмической модели — инструктивного описания процесса получения результата.

## 4.2. МОДЕЛИ И АЛГОРИТМ

Понятие «алгоритм» роднит математику и вычислительную технику. В математике термин «алгоритм» считается одним из древнейших. Он определяет ее высокую описательную мощь, позволяющую строить достаточно четкие модели самых разнообразных по своей природе фундаментальных научных понятий. Такие уточненные модели, будучи однажды построенными, затем становятся доступными для рассмотрения и совершенствования с помощью точных инструментов, и в изучении описываемых ими понятий наступает прогресс. Именно таким путем возникли математическая экономика, математическая лингвистика, математическая теория связи и другие разделы знаний. Математика, рассматриваемая с такой точки зрения, выступает в роли языка науки, задающего формальный способ описания, или алгоритм, решения задачи.

Математическое описание требует не только фиксации закономерностей исследуемого явления формулами или логическими выражениями, но и строжайшего ограничения области определения этого явления. (Вы помните это из школьной математики: то, что справедливо, например, для действительных чисел, необязательно верно для целых.) Тогда математическое описание будет гарантировать правильные ответы при решении поставленной задачи для любого из объектов, входящих в область определения. Именно такого рода уверенность в правильности решения конкретной задачи при использовании достаточно абстрактного описания и создает математике репутацию универсального языка науки и даже иногда «единственной науки». Однако не стоит понимать это буквально.

Во-первых, чтобы математика действительно давала правильные решения, необходимо всегда оставаться в рам-

ках области определения конкретного математического метода (или аксиоматики), а это крайне сложно при формулировке задач. Именно выход за пределы области определения, как правило, приводит к созданию новых, более универсальных методов. Известная Вам теория Ньютона вполне справедлива и сегодня, но для макрообъектов с ограниченной точностью представления. Теория Эйнштейна не опровергла теории Ньютона, а уточнила область ее определения макрообъектами и доказала ее противоречивость для микрообъектов. Новая область определения потребовала нового математического описания.

Во-вторых, помимо выявленных закономерностей, для которых удалось найти математическое описание, каждая область исследования содержит значительно больше не-систематизированной, интуитивно используемой, но не формализованной информации.

Таким образом, при оперировании математическими описаниями требуется четко представить, что, каким образом и для какого результата формализовано. Ясное понимание того, что Вы получаете в ответе, не менее важно, чем четкое ограничение области определения. С этой точки зрения математика — это инструмент в руках того, кто им умеет пользоваться, кому он помогает разобраться в сложной ситуации.

Для ЭВМ так же, как и для математики, алгоритм — это фундаментальная основа, определяющая описательную мощность и массовость действия. Остановимся подробнее на принципиальных аспектах алгоритмизации в связи с ЭВМ.

Алгоритм — это своего рода предписание, точным образом определяющее способ переработки исходных данных в результат. Требования к алгоритму, ориентированному на человека, несколько менее строги, нежели к алгоритму, предназначенному для ЭВМ. Это объясняется тем, что человек в отличие от ЭВМ постоянно привлекает, не отдавая себе в этом отчета, интуитивные знания и навыки в области своей деятельности. Машина этого не умеет.

Наиболее важные принципы алгоритмизации достаточно просты. Они состоят в следующем.

Во-первых, массовость. Если Вы, решая конкретную задачу, заранее знаете, что никогда не придется решать подобную ей, то не станете искать способа ее абстрактного описания, тем более что это и невозможно сделать, не определив хотя бы еще одной «аналогичной» в каком-то смысле задачи. Ведь и для выполнения индивидуального заказа нерентабельно строить специальный крупносерийный за-

вод. Другое дело, если Вы ожидаете, что некоторая задача будет многократно повторяться при различных исходных данных. Тогда абстрактное описание может существенно ускорить решение, причем уже не понадобится длительного анализа постановки задачи. Например, благодаря работам античных математиков мы располагаем удобными способами сложения, вычитания, на разработку которых ушли века. Принцип массовости тесно связан с заданием области определения и области решения задачи, т. е. предполагает строгое их задание, гарантирует правильное решение для любого объекта из ограниченной области и указывает способ интерпретации решения.

Во-вторых, детерминированность, или определенность. В соответствии с этим принципом требуется строгое определение каждого промежуточного шага к достижению решения. Если речь идет о решении задачи на ЭВМ, то под «строгим определением» понимается однозначное, не содержащее никаких двусмысленностей описание, единственно доступное ЭВМ.

В-третьих, эффективность, или результативность. Принцип эффективности означает, что в алгоритме должно быть предусмотрено условие окончания его работы. Например, если Вы хотите получить с помощью ЭВМ простые числа и забудете указать их количество или максимальное из них, то ЭВМ будет выдавать их до тех пор, пока не придет в негодность из-за износа, если, конечно, Вы сами не отключите ее.

Понятие «алгоритм» в интуитивном смысле успешно использовали на протяжении столетий. Считалось, что оно не требует уточнения. Слово «алгоритм» очень древнее и представляет собой латинскую транслитерацию имени великого среднеазиатского математика Мухаммеда бен Мусы аль-Хорезми (787 — ок. 850). Первоначально оно означало правило выполнения арифметических действий с использованием арабских цифр. На рубеже XIX—XX вв., когда многие математики занялись уточнением основных понятий арифметики, геометрии, алгебры и логики, в поле их зрения попал и алгоритм. В 30-х годах нашего столетия появилось около двух десятков определений алгоритма. Потом было доказано, что все они эквивалентны. А затем крупнейший советский математик А. Н. Колмогоров построил схему уточнения этого понятия, с помощью которой можно вводить много новых уточнений. Одно из них, сделанное А. Тьюрингом, наиболее близко к технической реализации. В связи с этим рассмотрим пример описания алгоритма для машины Тьюринга.

Прежде всего, как мы уже отмечали, машина Тьюринга — это абстрактная модель, не имеющая иного, кроме моделирования, отношения к технике. А. Тьюринг так описывает свою машину:

«Машина состоит из трех частей: запоминающего устройства, исполнительного устройства, устройства управления.

Запоминающее устройство — это склад информации. Оно соответствует бумаге, имеющейся у человека-вычислителя, независимо от того, является ли эта бумага той, на которой производятся выкладки, или той, на которой напечатана книга правил. Поскольку человек-вычислитель некоторые расчеты производит в уме, часть запоминающего устройства машины будет соответствовать памяти вычислителя.

Исполнительное устройство — это часть машины, выполняющая разнообразные индивидуальные операции, из которых состоит вычисление.

«Книга правил», заменяемая в машине некоторой частью запоминающего устройства, называется таблицей команд. Обязанность устройства управления — следить за тем, чтобы эти команды выполнялись безошибочно и в правильном порядке.

Информация, хранящаяся в запоминающем устройстве, разбивается на небольшие части, которые разбиваются по ячейкам памяти. Тем ячейкам, в которых хранится информация, приписываются номера в некотором порядке. Типичная команда: число, хранящееся в ячейке 6809, прибавить к числу, хранящемуся в ячейке 4302, а результат поместить в ту ячейку, где хранилось последнее из чисел. Такую команду можно было бы закодировать, например, в следующем виде:

6809

4302

17.

Здесь 17 представляет номер операции, а именно сложение, которое нужно проделать над числами, хранящимися в указанных ячейках.

Устройство управления выбирает команды в том порядке, как они расположены, но иногда могут встретиться команды типа: «выполнить команду, хранящуюся в ячейке 5606, и продолжать дальше» или «если в ячейке 4505 хранится 0, выполнить команду, хранящуюся в ячейке 6707, в противном случае продолжать по порядку».

Команды этого типа (безусловный и условный переход) позволяют повторять снова и снова некоторую последовательность действий до тех пор, пока не будет выполнено некоторое условие.

Поскольку мы имеем дело не с конкретной машиной, а с умозрительной абстракцией, легко представить себе память в виде ленты неограниченной длины.

Работа машины Тьюринга состоит в изменении состояния в зависимости от того, что подали на вход и каково текущее состояние, т. е. в переключении дискретных элементов по некоторым правилам».

Одной из важнейших характеристик алгоритма является время его выполнения, обычно исчисляемое в количестве шагов, приводящих к результату.



#### 4.3. ПРИЕМЫ ПОСТРОЕНИЯ АЛГОРИТМОВ

В алгоритмизации уже накоплен некоторый опыт, позволивший обобщить ряд полезных приемов построения алгоритмов для отдельных классов задач. Пополнение набора приемов алгоритмизации, по-видимому, приведет к тому, что процесс алгоритмизации постепенно от интуитивного творчества придет к систематической дисциплине. Перечислим некоторые наиболее употребимые приемы составления алгоритмов.

«Разделяй и властвуй» — этот способ применим в тех задачах, где исходные или промежуточные данные можно разделить на части, а общее решение получить из результатов вычислений по частям. Матричные операции сложения и вычитания выполняются этим способом, поскольку итоговая матрица составлена из вычисленных отдельно элементов. Простым примером применимости этого приема служит вычисление площади по частям. Так, неправильный многоугольник можно разбить на треугольники.

Когда известно приближенное решение и существуют способы его уточнения, можно использовать прием последовательных приближений. Начиная с некоторого известного решения  $f_0$  производится уточнение решения  $f_1$ , затем  $f_2$ ,  $f_3$  и т. д. Этот ряд представляет собой последовательность приближенных значений, которые или сходятся к действительному решению, или приближаются к нему настолько, насколько это нужно для практического использования. Так, число  $\pi$  можно вычислять по площади многоугольника, вписанного в единичную окружность: на каждом шаге вычислений надо увеличивать количество вершин до тех пор, пока не будет достигнута необходимая точность представления числа.

Название «наискорейший спуск» отражает заложенный в этом приеме смысл: для того чтобы достигнуть дна, требуется продвигаться только вниз. Следует отметить, что движение вниз не гарантирует достижения самой низкой точки. За дно можно принять первое обнаруженное углубление.

Обратный проход применяется тогда, когда задан порядок (направление) решения некоторой задачи; замена этого направления на обратное способна упростить задачу без ее изменения. Такой прием позволяет решать некоторые головоломки. Например, сколько прыжков понадобится сделать лягушке, чтобы выбраться из ямы глубиной

1 м, если за один прыжок лягушка поднимается на 30 см, а между прыжками сползает на 20 см? Проследивая путь лягушки в обратную сторону, находим, что она сможет выпрыгнуть из ямы, поднявшись на 70 см. Для этого ей потребуется семь прыжков, а на восьмом прыжке она выберется наружу.

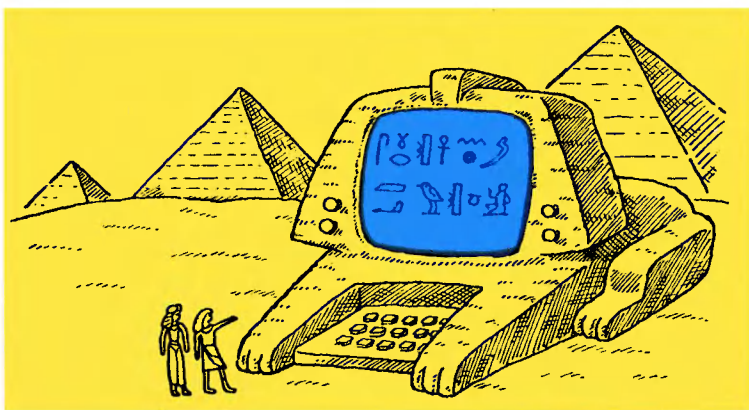
Идея динамического программирования заключается в одновременном использовании прямого и обратного проходов для решения задачи. Перемещение в одном направлении позволяет получить одно из возможных решений, движение в другом направлении обеспечивает альтернативное решение. После их сравнения выбирается лучшее решение.

Поиск с возвратом применяют, когда нет иного способа решения задачи кроме проб и ошибок. От некоторого решения осуществляют возврат к той точке, где был выбран один из альтернативных вариантов. После этого избирают путь по другому варианту, получают еще одно решение и выбирают лучшее. Затем осуществляют следующий возврат, и так до тех пор, пока не будет найдено удовлетворительного решения или кончится перебор вариантов. Этот прием лежит в основе многих программ для ЭВМ, играющих в шахматы.

Прием выделения подцелей заимствован из математики, где доказательство некоторой теоремы  $A$  осуществляется следующим образом: доказательство  $A$  вытекает из истинности  $B$ , доказательство  $B$  — из истинности  $C$ , доказательство  $C$  — из истинности  $D$ . Теорема  $D$  может быть доказана. Реальное доказательство проводится от  $D$  к  $C$ , потом к  $B$ , затем к  $A$ .

#### 4.4. ИСКУССТВО ПРОГРАММИРОВАНИЯ

Сегодня труд в любой области деятельности носит коллективный характер. Программирование не является исключением. Сочетание коллективного и индивидуального начал в труде программиста может служить мерилем профессионального умения. Было время, когда каждый программист писал собственную программу полностью для конкретной задачи. Этот путь привел бы к необходимости огромного числа программистов и к невысокой эффективности их труда. Оказалось полезнее обобщать наиболее употребимый программный продукт для коллективного пользования. Здесь, как нигде, ощутима общность между специалистами, в том числе



опосредствованно через программы и каналы связи во времени и пространстве. Мы можем не знать, кто автор той или иной программы или процедуры, но независимо от этого будем чувствовать его поддержку и помощь. В каждой программе незримо присутствует почерк, стиль автора. В этом смысле труд программиста индивидуален. Программист в своем конкретном деле стоит как бы на стыке творческой деятельности разработчиков программного обеспечения — его предшественников — и будущих пользователей.

Высокая ответственность программистов за результаты своего труда не в последнюю очередь продиктована колоссальной производительностью ЭВМ, тиражированием текстов программ. И если мы говорим, что «программирование — это вторая грамотность», то профессиональный программист — проводник этой грамотности. Именно эта профессия обещает стать продуктивным источником изменений в словарном составе языка науки и техники, что найдет свое отражение в общенародном современном русском языке и литературе.

Для успешного взаимодействия с техническими средствами и рационального их использования человек должен овладеть определенной системой знаний, приобрести соответствующие навыки, умение и опыт. При вождении автомобиля требуются быстрая реакция, хорошая пространственная ориентация, способность прогнозировать развитие дорожной обстановки в конкретной ситуации.

В чем же заключается специфика работы с таким совершенным инструментом, как ЭВМ? Какие качества наиболее важны для программиста-профессионала? Чтобы по-

нять это, нужно разобраться в главных методах работы программиста.

Основным методом, используемым в науке, является, как правило, анализ, в то время как практика идет по пути синтеза. Программирование — один из редких видов деятельности, в котором оба метода органически сочетаются. Программисту постоянно требуется выбирать из большого числа вариантов построения программ наилучший на всех возможных уровнях — от машинного слова до блоков и процедур. Одновременно он должен представлять, как и какими затратами достигается поставленная цель. И здесь специалисту особенно необходимо иметь развитую логическую память, прежде всего потому, что самые сложные и разнообразные ситуации должны быть описаны в виде линейной последовательности команд в ЭВМ. Для программиста крайне важны способность к обобщению, конструктивность мышления, точное восприятие обстановки, осознание ограничений со стороны технических и программных средств, подобно тому как водителю важно ощущать габарит автомобиля при въезде в узкий переулок.

Программист соотносит время работы программы и объемы занимаемой памяти. Всегда существует возможность увеличивать один ресурс с целью уменьшить другой. От удачно найденного баланса зависит эффективность последующего использования программ.

Не менее значима для программиста развитая структурная память — способность помнить большое количество фактов, объектов, а главное, их связи в пространстве и во времени. Такой памяти свойствен высокий уровень абстракции, так как внутренний язык ЭВМ — язык чисел. Как правило, системные программисты вырабатывают в себе способность взаимодействовать с ЭВМ на языке чисел. Но системный программист — это лишь одна из специализаций в области программирования, причем уникальная.

Одной из наиболее распространенных сегодня специализаций в области программирования является прикладное программирование. Цель прикладного программиста — помочь специалисту той или иной отрасли деятельности (искусство, социология, медицина, торговля, транспорт и многие другие) формализовать свои знания и на этой основе создать программное обеспечение для решения стоящей перед ним конкретной задачи. В этом случае программисту, несомненно, требуются такие качества, как способность быстрого переключения с одной предметной области на другую, гибкость восприятия. В этом отноше-

нии роль прикладного программиста подобна роли журналиста, интервьюирующего человека с целью сбора, обработки и публикации актуальной информации. Работа эта отнюдь не проста. Ведь большинство специалистов выражают свои знания в форме, весьма далекой от той, которая пригодна для представления в ЭВМ.

Особенно нужны в наши дни программисты, которые составляют программы для специализированных ЭВМ, управляющих станками с числовым программным управлением в условиях гибкого автоматизированного производства, программисты по обработке и хранению больших баз данных и связей между ними. В недалеком будущем объектами деятельности программистов, оперирующих графическими образами, явятся проектирование, дизайн, мультипликация. Элементы программирования необходимо освоить, конечно, любому человеку для возможности использования персонального компьютера — добросовестного помощника в повседневной работе и хорошего партнера в игре во время досуга.

Широкое использование ЭВМ не означает, что программирование станет самой распространенной профессией. Во-первых, каждый специалист, обучившийся элементам программирования, в большинстве случаев сможет обходиться без помощи программиста-профессионала. Во-вторых, процесс составления программы можно автоматизировать, если существуют методика и однозначный способ решения задачи. Несомненно одно — в ближайшее время каждому специалисту потребуются знания основ применения ЭВМ.

Именно программирование определяет эффективность использования ЭВМ, доминируя над ее аппаратными средствами. Труд программиста всегда находится на грани искусства, имеет большой творческий потенциал. ЭВМ, активно берущая на себя долю рутинной работы, освобождает людей для решения новых актуальных задач.

## 5 ПРОГРАММНЫЕ СРЕДСТВА

### 5.1. ЭЛЕМЕНТЫ ПРОГРАММИРОВАНИЯ

Каковы бы ни были причины, вызвавшие создание того или иного языка, язык программирования всегда должен отражать способ мышления специалиста. Это столь же, если не более, существенно, как и единство принципов архитектуры ЭВМ. В недолгой истории вычислительной техники периоды увлечения универсальными языками программирования, как правило, сменялись разработкой простых специализированных языков.

Следует отметить, что знание программирования напрямую не связано с добросовестным изучением конкретного языка. Скорее, это умение пользоваться наиболее общими для всех языков программирования структурными единицами. К ним относятся переменная, константа, цикл, условный оператор, процедура. Эти структурные единицы составляют основу практически любого языка программирования. В связи с этим рассмотрим кратко каждую из них, а затем обратимся к наиболее употребительным языкам высокого уровня, в которых они в том или ином виде используются.

Текст программы записывается и воспринимается последовательно, т. е. он состоит из четко различных элементов, следующих один за другим. Исполнение таких элементов будет заключаться в осуществлении строго определенных заданных действий над объектами, на которые указывает интерпретация элементов данных. Синтаксические элементы такого рода называются операторами.

Таким образом, оператор — это синтаксически различимый элемент языка программирования, служащий для представления основных объектов языка программирования: деклараций, процедур, функций...

## 5.2. ПРОЦЕДУРЫ И ФУНКЦИИ

Процедура в программировании имеет огромное значение. Наиболее часто применяемые процедуры объединяются в библиотеки процедур, которые могут использовать затем все программисты. Опыт коллективного творчества программистов выражен так или иначе в этом виде. Упрощенно можно сказать, что процедура — это программа, написанная заранее для последующего использования и реализующая ту или иную функцию (способ действия) для достижения некоторой промежуточной цели в решении всей задачи.

Так, в программе A1 по вычислению последовательности значений числа  $\pi$  необходимо получить площадь многоугольника, вписанную в единичную окружность. Для этой цели написана функция-процедура  $S(N)$ . Переменные с именами  $X1, X2, Y1, Y2$  соответствуют обозначениям в формуле для  $S_N$  следующим образом:

$$X1 = X_{i-1}, X2 = X_i, Y1 = Y_{i-1}, Y2 = Y_i.$$

Операторы  $X1 = X2$  и  $Y1 = Y2$  в цикле вычисления  $\sum_{i=2}^K (X_{i-1} - X_i) \times (Y_{i-1} + Y_i)$  производят присвоение координатам  $(i-1)$ -й точки значения координат  $i$ -й точки, т. е. выполняется перенос переменных соответственно увеличению  $i$  на единицу — оператор  $i = i + 1.0$ .

Такое переприсвоение значений переменных при изменении переменной цикла называют обычно рекуррентным шагом.

Методологически процедурный аппарат в программировании выражает более общий принцип декомпозиции (разделения) задачи на некоторое число более простых задач. И если полученные задачи еще сложны в реализации, то декомпозицию можно продолжать до тех пор, пока не будет достигнут ясный для человека уровень понимания задачи. Таким образом, образуется иерархия задач (целей) и, соответственно, иерархия процедур. Используя такой метод, можно решать довольно сложные задачи и получать надежный результат.

Заметим, что в программе A1 имеется только одно обращение к функции  $S(N)$ . Здесь выделение из программы операторов, вычисляющих площадь, вызвано только тем, что согласно постановке задачи вычисление площади является отдельной подзадачей. Такое построение текста программы делает ее легко читаемой и позволяет на контрольных примерах проверить  $S(N)$  отдельно от основной программы A1. Кроме того, использование процедур существенно экономит текст программы: так, в  $S(N)$  есть три обращения к процедуре  $COORD(X, Y, I, N)$ . Эта процедура вычисляет координаты  $X$  и  $Y$   $I$ -й точки  $N$ -угольника, вписанного в окружность.

C S - функция вычисления площади N-угольника,  
C заданного координатами вершин (X,Y) , по формуле:

C  
C  
C  
C  
C  
C  
C  
C  
C

$$S = 2 * \left( \sum_{I=1}^{K-1} (X_I - X_{I+1}) * (Y_I + Y_{I+1}) \right) + (X_K - X_1) * (Y_K + Y_1)$$

K=N/4+2  
I=2

```

DOUBLE PRECISION FUNCTION S(N)
DOUBLE PRECISION X1,Y1,X2,Y2,XK,YK
REAL N,K,I
K=N/4.0+2.0
CALL COORD(X1,Y1,1,0,K)
CALL COORD(XK,YK,K,K)
S=(XK-X1)*(YK+Y1)
I=2.0
1      CALL COORD(X2,Y2,I,K)
      S=S+(X1-X2)*(Y1+Y2)
      X1=X2
      Y1=Y2
      I=I+1.0
      IF(I.LE,K) GOTO 1
      IF(S.LT.0) S=-S
      S=2*S
      RETURN
      END

```

Функция S(N) вычисления площади K-угольника

С введением процедур в языки высокого уровня процесс программирования уподобился языковому творчеству. Каждая из процедур представляет некоторое понятие, а набор процедур на их уровне обобщения составляет тер-

```

SUBROUTINE COORD(X,Y,I,N)
DOUBLE PRECISION X,Y
REAL I,N
IF(I.EQ.1,0) GOTO 1
IF(I.EQ.2,0) GOTO 2
IF(I.EQ.N) GOTO 3
X=(I-2.000)/(N-2.000)
Y=DSQRT((N+I-4.000)*(N-I))/(N-2.000)
RETURN
1      X=0.000
      Y=0.000
      RETURN
2      X=0.000
      Y=1.000
      RETURN
3      X=1.000
      Y=0.000
      RETURN
      END

```

Процедура вычисления координат i-й точки K-угольника



минологический базис для более высокого уровня понимания задачи, на котором в свою очередь возможно образование новых процедур.

Этот процесс схож с созданием теории, в рамках которой производятся все большие обобщения. Наконец, будет получена законченная теория — программа, с помощью которой можно решить класс задач. При этом программа тем более ценна, чем шире этот класс задач.

Среди наиболее распространенных библиотек можно назвать библиотеки численного анализа и статистики, процедуры линейной алгебры и т. п. Как Вы заметили, это те процедуры, которые относятся к разделам классической математики. Процедуры, реализующие элементарные функции математики, включены в число основных средств (слов) языков программирования и называются встроенными функциями; к ним относятся  $\sin$ ,  $\cos$ ,  $\exp$ ,  $\sqrt{\phantom{x}}$  —  $\text{sqrt}$ .

В процедуре `COORD` имеется только одна встроенная функция `DSQRT`, которая извлекает квадратный корень из значения аргумента.

Термин «встроенная функция» программисты употребляют, когда функция реализована либо в виде процедуры библиотеки соответствующего транслятора, либо просто включена в состав системы команд процессора. Последнее означает, что функция вычисляется как одна команда ЭВМ специальной электронной схемой или микропрограммой процессора.

Но математика — не единственный источник широко известных библиотек процедур. Существуют библиотеки для обработки текстов, для работы с графическими объектами, изображениями.

Процедура оказалась удобным программным средством. Ее широко применяют на разном уровне коллективного пользования. Менее используемые и более объемные процедуры, например интегрирование, вводят в состав библиотек операционной системы. Кроме того, каждый программист может иметь личную библиотеку процедур. И наконец, в большинстве языков программирования предусмотрены средства описания процедуры в самой программе; это процедуры разового пользования.

Описание процедуры, как и любая другая программа, содержит описание последовательности действий. Различие процедуры и программы заключено в способе задания оператора именования. Этот оператор обязательно содержит формальные параметры, предназначенные для замещения конкретными значениями (фактическими параметрами) при каждом обращении к процедуре.

В А1 описание процедуры COORD начинается со строки 46, там же заданы формальные параметры X, Y, I, N. Переменные I и N содержат исходные данные, а в X и Y записывается результат работы процедуры.

В Алголе и Паскале для определения процедур принято слово *procedure*, в Фортране — *subroutine* и *function*. Обозначение процедуры в Фортране двумя терминами связано с общематематическим различием функции и отношения, т. е. при использовании слова *function* имя функции принимает значение результата вычислений. По этой причине в тексте программы вызова процедуры функции выглядят как некоторая переменная с аргументами. Кроме того, в языке Фортран существует оператор-функция. Эта конструкция очень похожа на математическое выражение, в левой части которого — имя функции с аргументами, а в правой — алгебраическая запись самой функции. Например, если для функции  $\sin(2x)$  выбрать имя  $\sin 2$ , а аргумент обозначить через  $x$ , то получим оператор-функцию  $\sin 2(x) = 2 * (\sin(x) * \cos(x))$ . Заметим, что  $\sin 2$  — это только имя, своим видом напоминающее алгебраическое выражение. Программисты именуют функции, сообразуясь с традицией и своими вкусами, а потому этот оператор мог бы выглядеть и так:  $W(x) = 2 * (\sin(x) * \cos(x))$ .

### 5.3. ПЕРЕМЕННЫЕ И КОНСТАНТЫ

Назначение переменной в программировании — фиксировать место в памяти под конкретную величину, принимающую разные значения, и впоследствии работать этой переменной по ее имени.

Вообще говоря, тексты программ и математические тексты имеют много общего по форме, но всегда следует помнить, что программы нацелены на действие, даже если это действие — передача сообщения.

Так, в математическом выражении

$$\sin 2x = 2 \sin x \cos x \quad (1)$$

$x$  называют переменной.

В выражении на языке программирования

$$A = 2 * \sin(x) * \cos(x) \quad (2)$$

$x$  также называют переменной. Однако значение переменной  $x$  в выражении (1) может быть любым, в то время как в выражении (2) переменная  $x$  имеет совершенно конкретное значение, а именно последнее значение, присвоенное

этой переменной. Пока переменной не присвоено новое значение, оно остается постоянным.

В программировании переменная величина понимается в конкретном, узком смысле, а в математике трактуется широко, вплоть до понятия «некоторая величина». Строго говоря, переменная в программировании — это символическое представление одного элемента памяти. Чтобы представить в ЭВМ такой математический объект, как вектор или матрица, ввели понятие индексированной переменной, которая определяется как символическое представление одного из группы элементов. При этом всю группу представляет имя, а индекс указывает на один элемент из этой группы.

Еще один частный случай переменной дает константа. Константа — это переменная, не подлежащая изменению в тексте программы. Как правило, имя константы выражает ее содержимое, например:

целое число 1, 216;

действительное 3.14159 или PI.

Однако следует иметь в виду, что в программировании часто встречается ошибка косвенного переопределения констант, например, при подстановке константы в роли фактического параметра на место формального. В том случае, когда этот параметр выступает в роли принимающего выходное значение из процедуры, будет изменено значение константы для всей программы, т. е. в других операторах будет использоваться новое значение вместо желаемого. Этот пример наглядно показывает различие в определении переменной в математике и программировании.

Процедуры, функции, переменные и константы образуют лексический багаж (словарь) программиста.

Включение в текст описания процедуры обращения к другим процедурам приводит к иерархии выполнения этих процедур. Такой вид организации программного текста напоминает, скорее, разъяснения одного понятия через другие, нежели задает структуру текста.

#### 5.4. ЦИКЛЫ И УСЛОВИЯ

В организации структуры текста программы наибольшую роль играют циклы и условия. Эта очевидная истина станет еще достовернее, если вспомнить построение текста программы на машинном языке, где циклический и условный процессы занимают ведущее место.

Цикл используется для описания операций, исполнение которых должно повторяться столько раз, сколько необходимо для достижения какой-либо определенной цели, или же столько раз, сколько это возможно без нарушения некоторого определенного условия. Эти два вида циклического оператора в языке программирования существенно различаются. Во-первых, цикл повторений должен осуществиться хотя бы один раз, в то время как цикл с предусловием (условием, определенным заранее) может быть пропущен, если условие цикла нарушено до его выполнения. Во-вторых, в цикле повторения выделена специальная переменная цикла для подсчета количества повторов, а цикл с предусловием не имеет такой переменной и опирается на соотношения переменных программы.

В тексте программы оператор цикла повторений в общем виде имеет следующую структуру: **ПОВТОРЯТЬ N РАЗ А**, где **А** — один или несколько следующих один за другим операторов, а **N** — количество исполнений **А**.

Существует и другая структурная разновидность цикла повторения, в которой непосредственно записывается условие (**У**) достижения цели, а не число повторов для ее достижения: **ПОВТОРЯТЬ А ПОКА НЕ У**. Такой цикл повторения иногда называют циклом с постусловием, поскольку условие **У** проверяется после исполнения **А**. В тот момент, когда **У** примет значение **ИСТИНА**, цикл завершается.

Структура оператора цикла с предусловием как бы развёрнута: **ПОКА У ВЫПОЛНЯТЬ А**. Здесь оператор **Р** исполняется, пока условие **У** имеет значение **ИСТИНА**. Оператор цикла завершится, как только **У** примет значение **ЛОЖЬ**; в частности, это может произойти сразу — тогда **А** вообще не выполняется.

Конечно же, эти два вида циклов взаимозаменяемы, и дело вкуса, каким из них чаще пользоваться. Но в любом случае во избежание ошибок необходимо помнить об этих двух пунктах различий цикла повторений и цикла с предусловием.

Предложение текста программы **ЕСЛИ У ТО В** называют условным оператором. Условие **У** определяет, следует или не следует исполнять оператор **В**. Смысл этого оператора будет яснее в контексте, где предыдущие операторы изменяют переменные так, что **У** принимает логическое значение **ИСТИНА** или **ЛОЖЬ**. Если **У** принимает значение **ИСТИНА**, то оператор **В** выполняется, что рассматривается как завершение всего условного оператора, и далее выполняется следующий в тексте оператор. В том

случае, если  $У$  — ЛОЖЬ, то выполнение условного оператора считается завершенным, и опять же выполняется следующий по тексту оператор.

Условный оператор можно рассматривать как частный случай альтернативного оператора: ЕСЛИ  $У$  ТО  $В$  ИНАЧЕ  $С$ .

В операторе этого вида  $В$  выполняется при  $У$  — ИСТИНА, а  $С$  — при  $У$  — ЛОЖЬ.

Условный оператор чаще всего используется для описания ситуаций, когда какое-либо действие надо либо исполнять, либо не исполнять. Альтернативный оператор применим при наличии различных ситуаций, требующих разного образа действий.

Отметим еще один элемент языков программирования: ВЫБОР  $У$  ИЗ  $A_1, A_2, \dots, A_N$ . Подобную конструкцию часто называют оператором выбора и нередко считают обобщением альтернативного оператора. Смысл оператора выбора заключается в исполнении одного из списка операторов  $A_1, A_2, \dots, A_N$ . Исполнение оператора выбора начинается с вычисления  $A_i$  значение которого определяет положительное целое число  $1 \leq i \leq N$ . Далее выбирается и исполняется  $A_i$ . Однако большинство употреблений этого оператора вызвано желанием побыстрее объединить ранее написанные части программы и в целом не обеспечивает существенных преимуществ с позиций компактности и ясности текста программы.

Оператор выбора в отличие от рассмотренных ранее операторов носит несколько искусственный характер. Дело в том, что переменная  $i$  вычисляется из выражения  $У$  в соответствии с подготовленными предыдущими операторами состояниями переменных программы. Таким образом, в действительности выбор происходит до исполнения оператора выбора и реализуется путем присвоения  $i$  подходящего значения. А это означает возможность исполнять один из списка операторов  $A_1, A_2, \dots, A_N$ , там, где уже подготовлено соответствующее состояние переменных программы.

Существенно необходимым оператор выбора становится в тех случаях, когда он наследует значение  $i$  из переменных, полученных программой извне. Иначе говоря, когда  $i$  подготавливает вызывающая программа или человек во время диалога с такой программой.

В других случаях использование оператора выбора приводит к неэкономному использованию оперативной памяти, а потому его по возможности следует избегать.

## 5.5. СТРУКТУРЫ ДАННЫХ

В большинстве языков программирования тексты программ содержат части, посвященные описанию данных. Это так называемые объявления, или декларации, данных. Они служат нескольким целям:

- именование переменных;

- описание типов данных, используемых в рассматриваемой программе;

- установление соответствия между переменными и типами данных;

- описание соответствия между структурами данных и структурами памяти ЭВМ.

Тип данного, конечно, не является чем-то особенным, присущим только языкам программирования. Роль типов данных подобна понятию «единицы физических величин», где возможность использовать в формулах ту или иную величину зависит от того, что это: сила тока, длина или атомная масса. Или, например, только последовательностью шести цифр 197101 будет до тех пор, пока не соотнесется с какой-либо числовой системой, в которой эти цифры определенным образом интерпретируются и служат объектом операций: если это почтовый индекс, то станет возможной почтовая операция адресации. Эти же цифры могут быть номером телефона, багажной квитанции и т. д.

Точно так же и в языках программирования одно и то же двоичное число может быть использовано с разными целями и различным набором операторов языка программирования. Посмотрите на таблицу «Простые типы данных». В ней есть строки с одним и тем же внутренним представлением в двоичном коде, и только тип данного определяет смысл нулей и единиц машинного слова.

И трансляторы, и процессор, и устройства ввода — вывода, работая с данными, интерпретируют их соответственно типу данного. Для этой цели в их составе имеются отдельные части, которые выполняют функцию интерпретации. В свою очередь на основе простых типов современные языки программирования предоставляют программисту возможность организовать более сложные типы данных, что не только повышает мощность языков, но и приближает структуру программы к структуре решаемой задачи.

Проектирование структуры данных в программе столь же важно, и занятие это, возможно, более сложное, чем написание собственно исполнительных операторов. Эта

## ПРОСТЫЕ ТИПЫ ДАННЫХ

ТИП ДАННОГО	ВНЕШНИЙ ВИД	ВНУТРЕННЕЕ ПРЕДСТАВЛЕНИЕ
НАТУРАЛЬНОЕ ЧИСЛО	124	000000000111100
ЦЕЛОЕ ЧИСЛО	-124	111111110000100
ДЕЙСТВИТЕЛЬНОЕ ЧИСЛО	124,0	011110000000000 0000000000001110
МНОЖЕСТВО	3,4,5,6,7	000000000111100
СИМВОЛ	Э	000000000111100
СТРОКА	ЭВМ	011110001110111 0110110100100000

сложность складывается из двух частей. Во-первых, всегда полезно определить структуры данных до написания текста программы, и, во-вторых, еще не многие языки программирования достаточно приспособлены для описания типов данных и структур данных.

### 5.6. ЯЗЫКИ ВЫСОКОГО УРОВНЯ

Сегодня насчитывается свыше 500 (!) различных языков для записи программ. Еще больше число вариантов (диалектов) этих языков. Для сравнения укажем, что в СССР представлено около 130 языков коренных народов. Этот факт легче всего объяснить специализацией профессиональной деятельности. Конечно, работа с чертежом существенно отличается от математических вычислений, а моделирование физических явлений и процессов — от ведения финансово-отчетной документации...

Однако только десяток языков послужил основой для создания почти всего программного обеспечения. Среди них языки, ориентированные исключительно на вычисление математических выражений, формул (Фортран, АПЛ), языки обработки списков (ЛИСП), да и просто учебные языки (Бэйсик, Паскаль и др.). Остальные 490 языков нельзя считать отходом, балластом в программировании. Каждый из них нашел свое применение, малый объем работ вовсе не означает их бесполезности.

Фортран — первый язык высокого уровня. Он был разработан в 1954—1957 гг. для программирования задач вычислительной математики, о чем свидетельствует и его название (*formula translation* — перевод формул). На этом чрезвычайно популярном языке решены многие как простые вспомогательные, так и сложнейшие задачи физики и других наук с развитым математическим аппаратом. На Фортране составлены самые обширные библиотеки научных программ. Накопленный опыт и действующие программы создают уникальные условия, благодаря которым Фортран еще долгое время будет активно использоваться, в то время как многие языки, получив широкую известность, спустя год-два исчезали из практики бесследно.

Наряду с Фортраном Алгол-60 (*algorithmic language* — алгоритмический язык) имел в свое время необычайную популярность. Его влияние, точнее, влияние конструкций и идей, заложенных в его структуру, до настоящего момента проявляется в программировании. Сегодня существует большое семейство алголоподобных языков. Столь значителен был успех разработчиков Алгола-60. И если сейчас этот язык не входит в первую десятку наиболее распространенных языков, то представители его семейства в ней есть и будут оставаться в ближайшее время. Основное назначение языка Алгол-60 — запись вычислительных алгоритмов математики в виде программ.

Язык Бейсик, разработанный в 1965 г. для учебных целей, существенно различается по способу трансляции команд. Как только машина получает законченное выражение, она тут же переводит его в коды, вычисляет и выдает результат. Этим режим работы особенно перспективен для вспомогательных расчетов компьютера, а потому Бэйсик является основным языком в программном обеспечении современного персонального компьютера. В Бэйсике каждая строка идентифицируется номером, на который можно ссылаться для управления вычислением. Центральное место в программе на языке Бэйсик занимают операторы FOR и NEXT, которые образуют цикл повторения всех операторов, находящихся в тексте между ними.

Активное программирование в таких областях, как экономика, где главная задача состоит в обработке данных с разнообразными структурами, обусловило появление новых языков. Прежде всего к ним относится Кобол (1960 г.). Разработка и довольно успешное использование Кобола продемонстрировали возможности языков программирования имитировать естественный язык профессиональной деятельности.



```

HPAL,"A1"
BEGIN
COMMENT A1 ВЫЧИСЛЯЕТ РЯД ПРИБЛИЖЕНИЙ К ЧИСЛУ ПИ ;
REAL PI,SI; INTEGER N;
FORMAT F1("PI=",F9,7,"    ПО ПЛОЩАДИ ",I8,"-УГОЛЬНИКА");
REAL PROCEDURE S(N);
COMMENT S = ФУНКЦИЯ ВЫЧИСЛЕНИЯ ПЛОЩАДИ N-УГОЛЬНИКА ;
VALUE N; INTEGER N;
BEGIN
REAL P,I,K;
P:=0; I:=3; K:=N/4+2;
WHILE I < K DO
BEGIN P:=P+SQRT((K+I-4)*(K-I)); I:=I+1 END;
S:=ABS(2*(1+2*P/(K+2))/(K+2))
END OF S ;
N:=4; PI:=1; SI:=0;
WHILE SI < PI DO
BEGIN SI:=PI; PI:=S(N); WRITE(6,F1,PI,N); N:=2*N END;
END OF A1 ;
END

```

PI=2,0000000	ПО ПЛОЩАДИ	4-УГОЛЬНИКА
PI=2,7320509	ПО ПЛОЩАДИ	8-УГОЛЬНИКА
PI=2,9957094	ПО ПЛОЩАДИ	16-УГОЛЬНИКА
PI=3,0898194	ПО ПЛОЩАДИ	32-УГОЛЬНИКА
PI=3,1232538	ПО ПЛОЩАДИ	64-УГОЛЬНИКА
PI=3,1351027	ПО ПЛОЩАДИ	128-УГОЛЬНИКА
PI=3,1392965	ПО ПЛОЩАДИ	256-УГОЛЬНИКА
PI=3,1407833	ПО ПЛОЩАДИ	512-УГОЛЬНИКА
PI=3,1413064	ПО ПЛОЩАДИ	1024-УГОЛЬНИКА
PI=3,1414881	ПО ПЛОЩАДИ	2048-УГОЛЬНИКА
PI=3,1415544	ПО ПЛОЩАДИ	4096-УГОЛЬНИКА
PI=3,1415815	ПО ПЛОЩАДИ	8192-УГОЛЬНИКА
PI=3,1415811	ПО ПЛОЩАДИ	16384-УГОЛЬНИКА

*Программа A1 на языке Алгол*

В отличие от Кобола язык АПЛ имеет крайне сжатый синтаксис, что позволяет писать очень короткие программы для сложных задач. Язык АПЛ так же, как и Бэйсик, ориентирован на режим интерпретации операторов. В руках квалифицированных программистов АПЛ становится удивительно удобным и превосходит по возможностям широко распространенные сейчас языки программирования. Несмотря на существующий барьер в восприятии АПЛ, язык получил довольно широкое распространение, и сегодня ожидается его включение в состав программного обеспечения персональной ЭВМ.

В 1970 г. для обучения информатике был разработан язык Паскаль. Этот язык необычайно быстро получил признание программистов в самых разнообразных отраслях. Распространению Паскаля способствует простота конструкций языка и, пожалуй, в наибольшей степени —

```

READY
LIST
010 PRINT " "
020 PRINT "ПРОГРАММА ВЫЧИСЛЯЕТ УТОЧНЯЮЩУЮСЯ"
030 PRINT "ПОСЛЕДОВАТЕЛЬНОСТЬ ЗНАЧЕНИЙ ЧИСЛА ПИ"
040 PRINT " "
050 LET N=4
060 LET P=0
070 FOR L=1 TO 16
080 LET S=P
090 GOSUB 170
100 PRINT "PI=";P;" ПО ПЛОЩАДИ ";N;"-УГОЛЬНИКА"
110 LET N=2*N
120 IF P <= 9 THEN 140
130 NEXT L
140 STOP
150 REMARK ПРОЦЕДУРА ВЫЧИСЛЯЕТ ПЛОЩАДЬ ВПИСАННОГО
160 REMARK В ЕДИНИЧНУЮ ОКРУЖНОСТЬ N=УГОЛЬНИКА,
170 LET K=N/4+2
180 IF K < 4 THEN 230
190 LET P=0
200 FOR I=3 TO K-1
210 LET P=P+SQR((K+I-4)*(K-I))
220 NEXT I
230 LET P=ABS(2*(1+2*P/(K-2))/(K-2))
240 RETURN
250 END
READY
RUN

```

ПРОГРАММА ВЫЧИСЛЯЕТ УТОЧНЯЮЩУЮСЯ  
ПОСЛЕДОВАТЕЛЬНОСТЬ ЗНАЧЕНИЙ ЧИСЛА ПИ

PI=	2	ПО ПЛОЩАДИ	4	-УГОЛЬНИКА
PI=	2,73205	ПО ПЛОЩАДИ	8	-УГОЛЬНИКА
PI=	2,99571	ПО ПЛОЩАДИ	16	-УГОЛЬНИКА
PI=	3,08982	ПО ПЛОЩАДИ	32	-УГОЛЬНИКА
PI=	3,12325	ПО ПЛОЩАДИ	64	-УГОЛЬНИКА
PI=	3,1351	ПО ПЛОЩАДИ	128	-УГОЛЬНИКА
PI=	3,1393	ПО ПЛОЩАДИ	256	-УГОЛЬНИКА
PI=	3,14078	ПО ПЛОЩАДИ	512	-УГОЛЬНИКА
PI=	3,14131	ПО ПЛОЩАДИ	1024	-УГОЛЬНИКА
PI=	3,14149	ПО ПЛОЩАДИ	2048	-УГОЛЬНИКА
PI=	3,14155	ПО ПЛОЩАДИ	4096	-УГОЛЬНИКА
PI=	3,14158	ПО ПЛОЩАДИ	16384	-УГОЛЬНИКА

READY  
BYE  
BASIC COMPLETED

*Программа A1 на языке Бэйсик*

возможность организации составных типов данных на основе простых. На базе Паскаля было разработано несколько более сложных языков, в том числе язык Ада.

Особое место среди языков программирования занимает ЛИСП. В нем есть только один вид инструкций — функция. Он обладает рекурсивностью: значением функ-

ции может быть любая другая функция. ЛИСП был создан в конце 50-х годов, но до сих пор остается одним из самых популярных языков при обработке данных, на что он и был исходно ориентирован (LISP — list processing — обработка списков). Для вычислительных задач использовать ЛИСП не столь удобно.

Все перечисленные языки относятся к языкам процедурного типа. Для них характерно императивное задание выполнить определенную последовательность действий.

Существуют также непроцедурные, или описательные (декларативные), языки, на которых задают некоторые соотношения и целевые установки, но не указывают, как конкретно достичь цели. Некоторую разновидность этого типа языков представляют объектно-ориентированные языки. В объектно-ориентированном языке ЭВМ условно делится на объекты, к которым можно обращаться индивидуально. Программный объект состоит из структур данных и алгоритмов. Каждый объект «знает», как выполнять операции со своими собственными данными, но для остальной программы неизвестна внутренняя организация объекта. Различные объекты могут пользоваться совершенно разными алгоритмами при выполнении заданий, определенных одинаково.

```
PROGRAM A1 (OUTPUT);
(* A1 ВЫЧИСЛЯЕТ РЯД ПРИБЛИЖЕНИЙ К ЧИСЛУ ПИ *)
VAR PI,SI :REAL; N :INTEGER;
FUNCTION S(N :INTEGER) :REAL;
(* S ВЫЧИСЛЯЕТ ПЛОЩАДЬ N-УГОЛЬНИКА *)
VAR P,I,K :REAL;
BEGIN   P:=0; I:=3; K:=N/4+2;
  WHILE I < K DO
    BEGIN P:=P+SQRT((K+I-4)*(K-I)); I:=I+1 END;
    S:=ABS(2*(1+2*P/(K-2))/(K-2))
  END (*S*);
BEGIN
  N:=4; PI:=0;
  REPEAT SI:=PI; PI:=S(N);
    WRITELN ('PI=',PI,' ПО ПЛОЩАДИ ',N,'-УГОЛЬНИКА');
    N:=2*N UNTIL PI<=SI
END (*A1*);
```

PI=1,9999999925494E+0	ПО ПЛОЩАДИ	4=УГОЛЬНИКА
PI=2,7320508845150E+0	ПО ПЛОЩАДИ	8=УГОЛЬНИКА
PI=2,9957094080746E+0	ПО ПЛОЩАДИ	16=УГОЛЬНИКА
PI=3,0898194238544E+0	ПО ПЛОЩАДИ	32=УГОЛЬНИКА
PI=3,1232538111508E+0	ПО ПЛОЩАДИ	64=УГОЛЬНИКА
PI=3,1351027395576E+0	ПО ПЛОЩАДИ	128=УГОЛЬНИКА
PI=3,1392965223640E+0	ПО ПЛОЩАДИ	256=УГОЛЬНИКА

*Программа A1 на языке Паскаль*

Итак, имеется много разных языков программирования, и их число все больше увеличивается. Не существует самого лучшего языка программирования, каждому из них свойственны свои преимущества в конкретных приложениях.

## 5.7. ПРОГРАММНОЕ ОКРУЖЕНИЕ

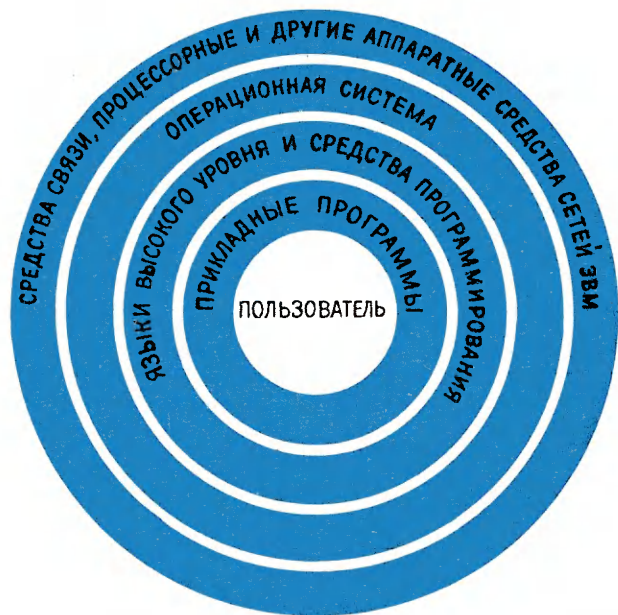
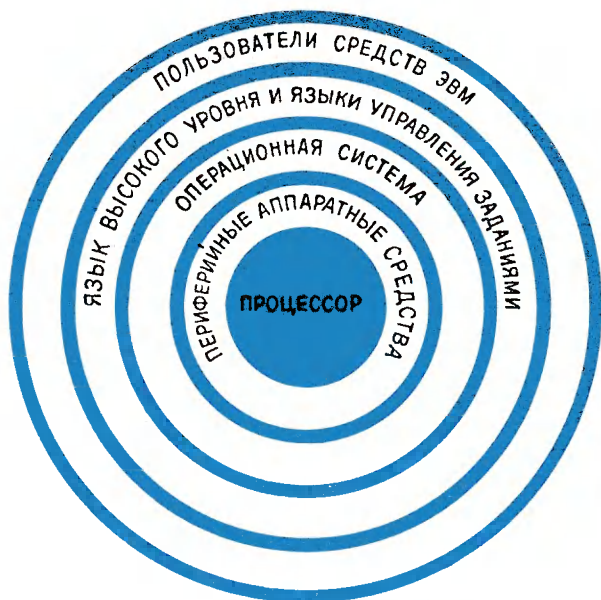
Программа неотделима от вычислительной среды, с которой она взаимодействует. Значительно проще создать хорошую программу, располагая такими эффективными вспомогательными средствами, как специализированные языки программирования, каталоги данных, трансляторы и т. д. Кроме того, существенно важно построить программу таким образом, чтобы ее можно было успешно применять в различных приложениях.

Дело в том, что аппаратные средства вычислительной техники дешевеют с каждым годом, а программное обеспечение имеет тенденцию к усложнению и удорожанию. Основная причина этого кроется в самих ЭВМ. Большой спрос на ЭВМ создает условия для их крупносерийного производства, в условиях которого проектировщики и изготовители ЭВМ, естественно, могут быстрее и основательнее внедрять автоматизированные технологические процессы. Программисты же всякий раз имеют дело с единственной в своем роде задачей. Средства автоматизации их труда охватывают процессы, связанные с преобразованием текстов, но пока не эффективны на этапах проектирования и контроля программного продукта.

На наших глазах происходит кардинальное изменение в отношении к вычислительной технике. Если всего 10—15 лет назад основное внимание уделялось загруженности процессора, ликвидации даже минутных его простоев, то сегодня в центре внимания находится пользователь с его интересами. Повсеместно стал внедряться принцип: во взаимодействии человека и ЭВМ ожидающей стороной может быть только ЭВМ!

Сейчас человек за терминалом сети вычислительных машин эксплуатирует несколько процессоров, тогда как раньше в распоряжении большого коллектива программистов имелся один процессор. Изменение режима взаимодействия человека и машины наиболее ярко прослеживается на примере развития операционных систем.

Первые операционные системы появились в конце 40-х годов и представляли собой наборы простых процедур для обеспечения операций ввода — вывода. Впрочем, в то



*Во взаимодействии человека с ЭВМ ожидающей стороной должна быть только ЭВМ!*

время их не отличали от других библиотек программ. В середине 50-х годов одновременно с повышением быстродействия ЭВМ потребовалось сократить интервалы между концом работы одной программы и началом работы другой. Операционные системы того времени обеспечивали исполнение программ одну за другой. Впоследствии такое обслуживание называли пакетным режимом. Кроме того, они обеспечивали доступ к внешней памяти. Это была уже довольно сложная программная система.

Десять лет спустя, к середине 60-х годов, были разработаны системы с разделением времени. В основу этого режима была заложена идея прерываний вычислительных процессов. Процессор переключался с одной программы на другую, создавая у пользователей впечатление одновременности работы с ЭВМ.

Тогда же началось активное внедрение мини-ЭВМ в управление технологическими процессами. Для реализации этих задач потребовались новые качества операционных систем, необходимость реагировать на внешние прерывания при жестком ограничении, накладываемом на время ответа. Эти системы называли операционными системами реального времени.

Сегодня операционная система — это организованная совокупность программ, которая действует как интерфейс между аппаратурой ЭВМ и пользователями. Она обеспечивает их набором средств для облегчения проектирования, программирования, отладки и сопровождения программ и одновременно управляет распределением ресурсов для эффективной работы программ.

Опыт по эксплуатации операционных систем в режиме плотной загрузки единственного, центрального процессора не пропал даром, не остался просто историческим этапом развития ЭВМ. Сегодня мы относимся к ресурсам супер-ЭВМ так же ответственно, как раньше к обеспечению максимальной загруженности центрального процессора. СуперЭВМ, эти вычислительные гиганты, находящиеся в центре вычислительной сети, требуют наибольшего внимания, поскольку именно они в конечном счете определяют мощность и возможности всей сети ЭВМ.

Итак, в развитии программного обеспечения ЭВМ и в первую очередь операционных систем наступил период, когда ведущие позиции стали занимать проблемы интерфейса пользователя. Понятие «интерфейс пользователя» включает прежде всего формы взаимодействия человека и ЭВМ. От степени удобства этих форм существенно зависит эффективность труда специалиста — пользователя

средств вычислительной техники. Под формами взаимодействия подразумевается не только вид получения или передачи информации, но и порядок следования отдельных строк, текстов, таблиц, графиков и т. п.

В интерфейс пользователя следует включить и носители форм взаимодействия человека и машины — дисплей, табло, индикаторные панели и пр. И наконец, наиболее важный компонент интерфейса пользователя — программное обеспечение, которое динамически организует всю аппаратуру терминального комплекса в соответствии с потребностями специалиста и ходом вычислений по прикладным программам.

#### 5.8. СРЕДСТВА ОБЩЕНИЯ С ЭВМ

Все, что производит ЭВМ, человек воспринимает через интерфейс пользователя, который выполняет роль посредника между человеком и прикладными программами, превращающими компьютер в инструмент для решения конкретной задачи. Сейчас интерфейсу придается первостепенное значение, поскольку и новичку, и квалифицированному специалисту компьютер представляется именно таким, каким он его ощущает через интерфейс. Это «представление» пользователя позволяет ему понимать действия системы и вырабатывать адекватные реакции.

Для персональных ЭВМ наиболее удобными оказываются так называемые системы поддержки решений, в которых имеется широкий выбор программ как вычислительного, так и вспомогательного характера (например, для организации диалога). Из них, как из кубиков, можно создавать огромное число разнообразных текстов программ. При этом процесс программирования вовсе не так сложен: нужно лишь производить обращение к соответствующим процедурам, подставляя конкретные значения параметров. Такой способ общения очень удобен в ходе обучения программиста и одновременно вполне достаточен для решения многих прикладных задач. Большое достоинство таких систем состоит в возможности пополнения набора процедур.

Сейчас в компьютеризованной системе проектировщик составляет первоначальный вариант проекта и передает его математику. Тот создает математическую модель объекта и выбирает или разрабатывает алгоритм ее решения. Следующим в цепочке является программист. Он пере-

водит математическую задачу на язык, понятный машине, и передает программу на ЭВМ. Результаты расчета проходят тот же путь в обратном направлении. И так происходит с каждым последующим вариантом проекта, в связи с чем на эту работу затрачиваются длительное время и большие усилия. Комплексный диалоговый интерфейс, заменяя промежуточные звенья, позволяет сократить цепочку до двух элементов — проектировщик и ЭВМ. Конечно, создать интеллектуальный интерфейс непросто: машине нужно передать тот объем знаний, которым владеет и программист, и математик, и даже отчасти сам проектировщик — иначе она его не поймет. Но зато, как показывает практика, качество проектирования при работе с комплексным интерфейсом повышается, а затраты труда для данного типа проекта сокращаются. Кроме того, появились дополнительные возможности и методы проектирования. До сих пор при всем разнообразии целей, проблем и технологических возможностей инженеры почти всегда решали прямые задачи — определяли характеристики выбранной конструкции. Если они оказывались удовлетворительными, инженерная мысль воплощалась в металле, если нет, разрабатывались новые варианты конструкции. С развитием методов оптимизации, математического аппарата и вычислительной техники посильным стало решение обратной задачи: по заданным характеристикам искать вид конструкции, синтезировать ее. Не следует думать, что с внедрением ЭВМ резко уменьшится интенсивность труда конструктора. По мере расширения масштабов компьютеризации, передачи ЭВМ рутинных процессов в работе конструкторов труд их станет более творческим и интересным, а эффективность его существенно возрастет.

Сейчас наиболее популярны такие приемы в обеспечении интерфейса, как окна, меню, таблицы. Разделив экран дисплея на кадры-участки, называемые окнами, можно выводить на экран информацию о ряде одновременно выполняемых процессов. Меню — это выдача на экран всех возможных продолжений работы, из которых пользователь выбирает на каждом шаге единственное.

Об удобстве представления информации в виде таблицы мы уже говорили ранее.

Такого рода средства интерфейса зачастую более наглядны, чем обычный текстовый диалог, так как позволяют одновременно охватить несколько объектов в их взаимосвязи, в противовес линейному просмотру тех же объектов. К тому же при поиске информации отпадает необходимость



в наборе на клавиатуре подробного запроса, что существенно снижает вероятность ошибки.

Принцип обозримости информации во всех взаимосвязях наиболее актуален при создании интерфейса. Ведь правильно истолковать решение не менее важно, чем его получить. А этому в значительной степени способствует простота внешнего представления решения. По этой причине одним из наиболее перспективных видов представления информации является крупноформатный электронный бланк (КЭБ) в виде прямоугольной матрицы на экране дисплея. Для каждой клетки существует свое правило нахождения значения. Если изменить значение какой-либо позиции, то все зависящие от нее позиции будут мгновенно скорректированы и воспроизведены на экране. Необычная простота работы с КЭБ сочетается с высокой надежностью и быстротой модификации данных. Кроме того, в КЭБ «естественности» гораздо больше, чем в ограниченном естественном языке в диалоге.

КЭБ — это первый шаг в общем направлении развития интерфейса с человеком, который в идеале представляется так: получив задание в общей формулировке, система самостоятельно выполняет его, в том числе выбирает метод вычислений, а в случае затруднений (нехватка информации или отсутствие метода решения) обращается с запросом к человеку.

Вообще говоря, способ организации общения человека с ЭВМ в наибольшей степени определяет широту его распространения. Чем больше разнообразных форм будет в арсенале вычислительной техники, тем вероятнее, что любой человек — как профессионал, так и любитель — сможет подобрать удобную для себя форму общения с ЭВМ, а значит, использовать ее эффективно, творчески.

## 6

# ИНТЕЛЛЕКТ ЕСТЕСТВЕННЫЙ И ИСКУССТВЕННЫЙ

### 6.1. МОЖЕТ ЛИ МАШИНА МЫСЛИТЬ?

Такой вопрос поставил А. Тьюринг, однако подразумевал при этом не обычную машину, а некоторую абстракцию, эквивалентную понятию «алгоритм» или «вычислительная функция». Вопрос А. Тьюринга по существу равносильен вопросу о том, носит ли мышление человека алгоритмический характер, нельзя ли представить его в виде некоторого универсального алгоритма.

А. Тьюринг особо отмечал, что, имея дело с машиной, программа которой неизвестна, открыть по «поведению» этой машины ее программу исключительно трудно. Рассмотрим в связи с этим эксперимент А. Тьюринга, который был назван «игрой в имитацию».

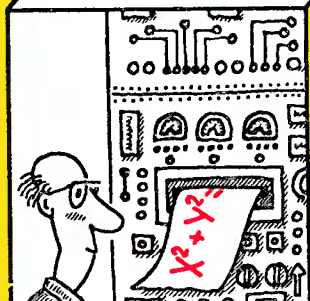
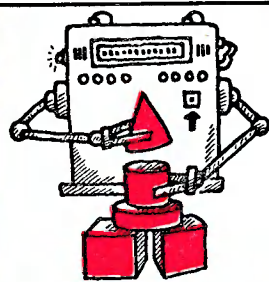
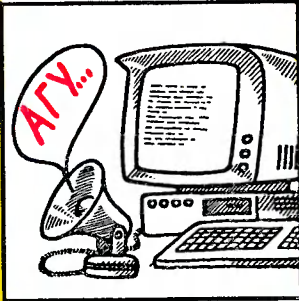
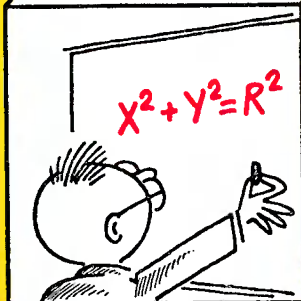
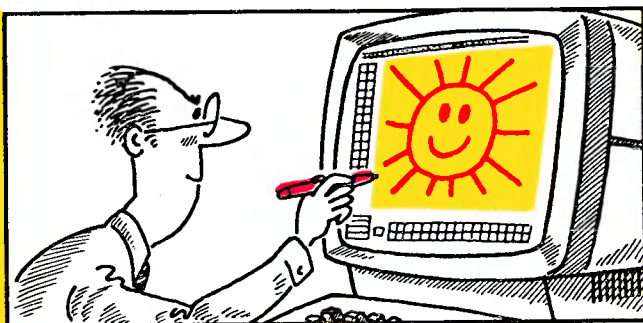
В игре участвуют три человека: мужчина *A*, женщина *B* и некто *C*, задающий вопросы и отделенный от двух других участников игры стеной. Цель игры для *C* — определить, кто является мужчиной, а кто — женщиной. Он знает их под обозначениями *X* и *Y* и в конце игры должен сказать, что «*X* есть *A*, *Y* — *B*» либо что «*X* есть *B*, а *Y* — *A*». Ему разрешается задавать вопросы такого рода: «Прошу *X* ответить, какова длина его волос».

Теперь допустим, что в действительности *X* есть *A*. Для *A* цель игры состоит в том, чтобы побудить *C* прийти к неверному заключению. В связи с этим ответ его может быть, к примеру, таким: «Мои волосы коротко острижены, а длина самых больших прядей около 2—3 см».

Разумеется, чтобы отвечающих нельзя было различить по голосу, диалог ведется письменно.

Цель игры *B* — помочь задающему вопросы. Для нее наилучшая стратегия — давать правдивые ответы. Но даже если она скажет: «Женщина — я!», — это мало поможет, поскольку и *A* может сделать подобное заявление.

В постановке такого эксперимента А. Тьюринг тщательно оговаривал, в чем именно устанавливается сходство между человеком и ЭВМ. Здесь нет речи о внешнем сходстве.



Путь к знанию человека и машины

А. Тьюринг описал этот эксперимент на заре вычислительной техники, когда машины и программы были еще очень несовершенными, когда лишь фантасты брали на себя смелость наделять бездушные машины антропоморфическими (человеческими) свойствами. В этом стремлении предугадать дальнейшее развитие и последствия научного открытия и состоит поступательное движение научно-технического прогресса.

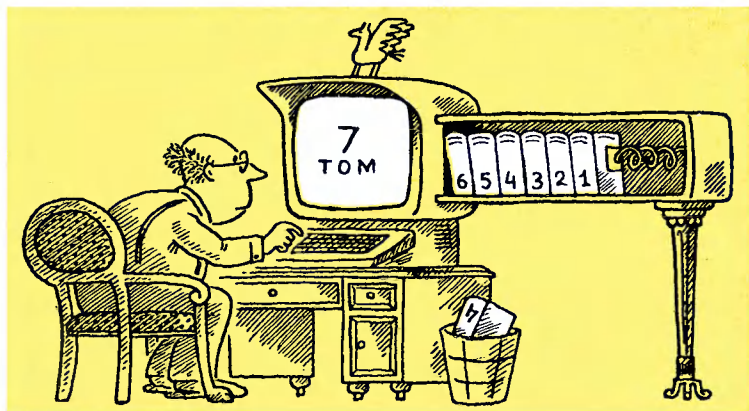
Подмена человека комплексом программ, реализуемых на ЭВМ, создает иллюзию, что в основе программ лежат те же процессы, которые протекают и в мозгу человека при решении аналогичных задач. Одинаковость результатов как бы свидетельствует об одинаковости процессов, приводящих к этим результатам. К сожалению, как только составлена программа для решения на машине какой-то задачи, то сразу возникает внутреннее сомнение в том, что эта задача творческая и требует для своего решения интеллекта.

Мы привыкли понимать под интеллектом нечто единое, позволяющее человеку тонко реагировать на изменения внешнего мира и приспосабливать свое поведение к этим изменениям. Но разве нельзя представить интеллект как целый спектр переплетающихся свойств, каждое из которых в отдельности отнюдь не вызывает особого восхищения, но при проявлении одновременно достаточно большого их числа поведение приобретает характер разумного? Целью такого подхода является создание систем искусственного интеллекта, т. е. выяснение возможностей ЭВМ по имитации той деятельности, которую мы склонны считать творческой и интеллектуальной, когда ее реализует человек.

Комплексная научно-техническая проблема создания искусственного интеллекта объединяет четыре направления.

Первое направление связано с имитацией на компьютере творческих процессов. Люди разрабатывают алгоритмы и программы игры в шахматы, анализа и синтеза музыкальных произведений, доказательств теорем, автоматического синтеза программ. Такие алгоритмы ориентированы на конечные результаты творческого процесса, а не на воспроизведение деятельности человеческого мозга.

Другое направление — внешняя интеллектуализация компьютеров, разработка диалогового интерфейса, т. е. системы общения с ЭВМ, доступной неспециалисту, — позволит не только вести прямой (без помощи программистов) диалог с ЭВМ, но и улучшить процесс принятия решений в математически слабоформализованных обла-



стях, таких, как биология, медицина, общественные науки, управление сложными народнохозяйственными объектами.

Журналисты, писатели, работники науки и техники, пройдя однажды весь путь подготовки текста на персональном компьютере, вряд ли предпочтут вернуться к обыкновенной пишущей машинке. Пользуясь персональной ЭВМ, легко вставить и изъять, переставить и объединить различные части текста. Можно одновременно работать с несколькими вариантами текста. ЭВМ проследит за выравниванием строк по ширине и ограничением их по высоте в соответствии с заданным форматом, а кроме того, укажет на ошибки в написании слов.

Применение вычислительной техники, скажем, в полиграфии коренным образом преобразует технологический цикл производства печатной продукции. Взяв на себя функции наборщика, ЭВМ как бы подталкивает человека к использованию машинного способа подготовки текста на всех этапах, предшествующих выпуску книги.

Не менее важна и внутренняя интеллектуализация ЭВМ — повышение производительности и улучшение характеристик компьютеров за счет совершенствования их архитектуры и разработки новых интеллектуализированных способов обработки информации. Совместная реализация двух последних направлений приводит к созданию ЭВМ пятого поколения.

И наконец, все большее значение приобретают моделирование целенаправленного поведения роботов и создание интеллектуальных роботов (роботов третьего поколения). Эта проблема особенно сложна, так как робот в отличие от обычной ЭВМ своими действиями

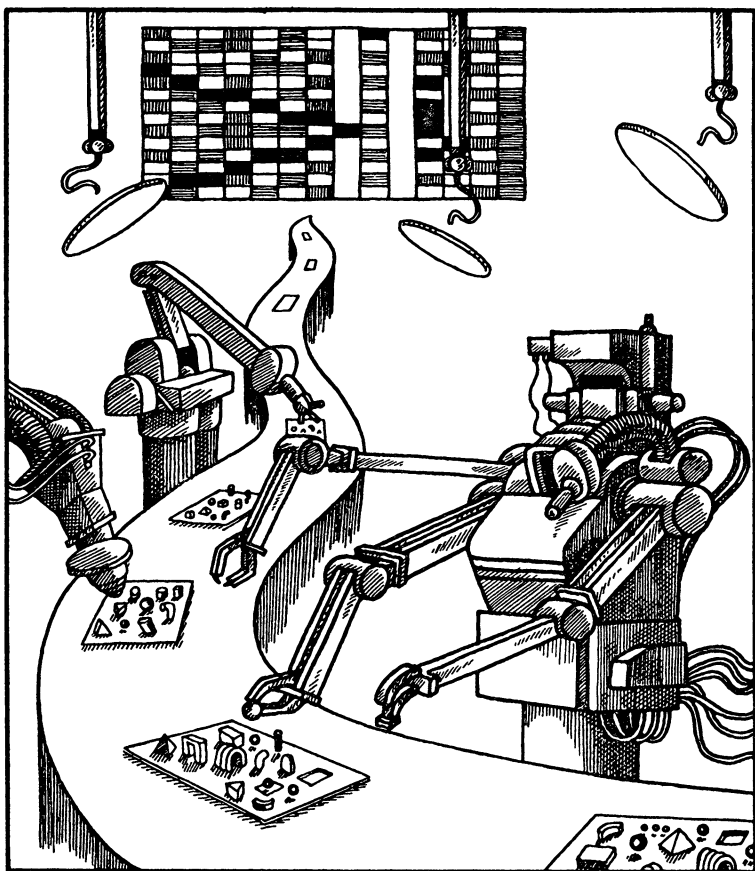
влияет на обстановку вокруг себя и ему приходится решать задачи с изменяющимися условиями, а методы решения таких задач еще не достаточно разработаны.

По мере усложнения технических систем последовательно создаются все новые помощники человека. И появление ЭВМ исторически закономерно, поскольку развитие технических систем всегда приводило к необходимости в более совершенных и гибких средствах их информационного описания (см. левую и правую ветви рисунка на стр. 8). Среди средств информационного обмена в обществе — начиная с передаваемых из поколения в поколение преданий и кончая необозримым потоком продукции полиграфической промышленности — ЭВМ выступает в роли качественно нового инструмента хранения, поиска, преобразования и распространения информации. Вот почему эта сильная сторона ЭВМ делает возможной разработку систем с искусственным интеллектом.

ЭВМ по праву можно назвать универсальным помощником во всех сферах человеческой деятельности. Постепенно от решения крупных проблемных задач (в области физики, автоматического управления, численного решения уравнений и т. д.) перешли к использованию ЭВМ в составе автоматизированных систем управления, а затем для управления работой конечного узла технологической цепочки производства. Именно разработка всех элементов технологической цепочки (манипуляторов, роботов, станков, складов, измерительных приборов и т. д.), управляемых встроенными микропроцессорами с помощью ЭВМ, позволила выдвинуть концепцию создания «безлюдного» производства.

## **6.2. МОЖНО ЛИ ОБОЙТИСЬ БЕЗ ЧЕЛОВЕКА?**

Гибкие производственные системы — новая форма технологической организации производства. В конечном счете научно-технический прогресс всегда приводил к перераспределению трудовых ресурсов. Механизация сельского хозяйства позволила производить больше продуктов при резком сокращении численности работающих. С внедрением конвейерной формы организации труда на производстве ряд технологических операций оказалось возможным полностью автоматизировать. Но при этом слабым звеном, сдерживающим рост производительности труда, явились узкоспециализированные работы, требующие специальной



*Гибкие производственные системы — новая форма технологической организации производства*

подготовки и профессиональных навыков,— токарные, фрезерные и т. д.

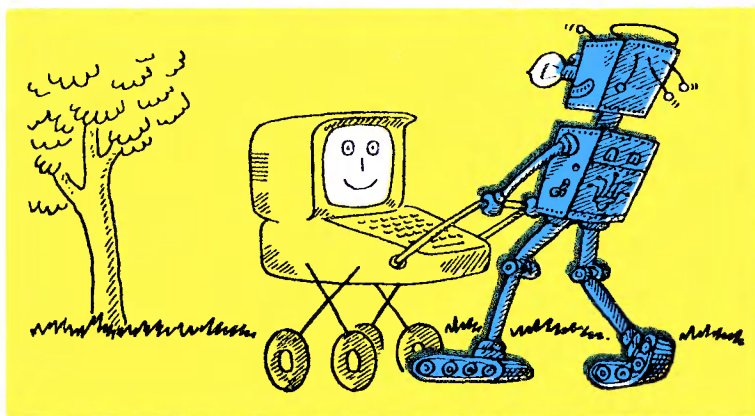
С появлением роботов, манипуляторов, станков с числовым программным управлением (ЧПУ), универсальных обрабатывающих центров изменился подход к организации производства. Прежде всего стало возможным тиражировать «профессиональные знания», так как программа для станка с ЧПУ позволяет оператору управлять одновременно несколькими станками. Следующим шагом на пути организации «безлюдного» производства явилось объединение функций конструктора, технолога и других участников производственного процесса благодаря

унификации и обмену программами и информацией посредством информационно-вычислительной сети.

Но одна из основных проблем заключается в создании замкнутой ячейки технологического цикла, полностью управляемой от ЭВМ, которая включает манипулятор — станок — транспорт — склад. Объединение ряда различных ячеек позволяет организовать единый технологический цикл изготовления изделий. В условиях гибкого автоматизированного производства необходимы только операторы, устраняющие неисправности по сигналам контроля. Но внешнее впечатление «безлюдности» производства обманчиво. На самом деле для его организации требуется труд большого числа специалистов, и особенность его состоит в том, что каждый специалист вместо изготовления, перемещения, конструирования... занимается о п и с а н и е м процессов изготовления, перемещения, конструирования...

Переход от непосредственного производства к опосредствованному — через описание — обеспечивает ЭВМ. Здесь воплощается важнейшая способность вычислительной техники — переводить описание в действие. Одновременно проявляется другая важная способность — выбирать оптимальное описание из многочисленных вариантов, хранимых в памяти ЭВМ. И наконец, побочным эффектом гибкой автоматизации производства является изменение стиля управления на основе безбумажной технологии.

Таким образом, вопрос «можно ли обойтись без человека?» следует уточнить: «без человека какой профессии и какого уровня подготовки?».





Совершенно очевидно, что современный уровень научных знаний имеет определенную границу. Один известный ученый на вопрос «почему Вы чаще говорите „не знаю“, нежели молодые специалисты?» ответил: «Я обладаю, конечно, большим объемом знания, чем они, но и граница моего незнания также больше».

При разработке искусственного интеллекта — передового направления современных исследований — наиболее ярко выявляется грань возможного и невозможного. Серьезные ограничения, связанные с понятием «сложность» применительно к ЭВМ, вытекают из невозможности иметь полное описание сложной ситуации. Согласно гипотезе Лапласа (1796), «если бы в один момент можно было точно узнать все положения и скорости всех элементов Вселенной, то можно было бы точно предсказать будущее». Недостижимость такой ситуации следует из соотношения сложности объектов и трудности их описания. Начиная с некоторого «порога сложности» описание объекта становится сложнее самого объекта. Сошлемся в связи с этим на мнение А. Н. Колмогорова: «Возможно, что автомат, способный писать стихи на уровне больших поэтов, нельзя построить проще, чем промоделировав все развитие культурной жизни того общества, в котором поэты реально развиваются».

Сейчас мы уже вплотную подошли к проблеме представления информации об огромном числе элементов со сложными взаимосвязями. Вообще эта взаимосвязанность постоянно интересовала человечество, но более или менее умозрительно, с точки зрения познания закономерностей природы. В настоящее время эта проблема переросла в сугубо практическую. Мы хотим знать, к каким последствиям приведет, например, осушение болота или уничтожение вредного насекомого. От установки на «покорение» природы, властвовавшей над умами до недавнего времени, мы переходим к пониманию необходимости самого бережного ее использования и восстановления. В решении таких проблем мы вправе рассчитывать на ЭВМ. Какой же еще инструмент может быстро и точно обработать огромное количество информации и подсказать правильное решение или, по крайней мере, уберечь от катастрофических последствий принятия неправильного решения? Но для решения столь глобальных задач, а сегодня мы уже вплотную к ним подошли, требуются чрезвычайно

разнообразные знания, представленные в самых различных формах. Многие из наших знаний сформулированы нестрого, мы зачастую полагаемся на интуицию, резко отвергая ту или иную идею.

ЭВМ сегодня уже полностью готова к решению очень объемных задач экономического или социального характера. Но необходимо объяснить машине, что именно мы хотим решить и какими сведениями располагаем. Вот тут-то и возникают сложности. Даже в общении людей считается, что постановка задачи — это половина ее решения. А когда в качестве партнера в общении выступает ЭВМ, можно смело сказать, что это уже почти решение. Современное быстродействие и емкость памяти ЭВМ гарантируют незначительные затраты времени на собственно вычисления и выдачу результата.

Процесс формулирования задачи — прерогатива человека. Машина может помочь и в этом процессе, взяв на себя выработку плана решения в рамках заданной стратегии, формулирование подзадач и некоторые другие вспомогательные функции. Но ответственность за достижение цели и выбор данных, существенных для решения, целиком возлагается на человека. И ответственность эта велика. Например, при исследовании экологической системы человек исходно предполагает, что осушение болота может повлечь за собой пересыхание некоторых рек. Он вводит все данные, связанные с параметрами рек, и получает положительный результат: реки не пострадают. Значит ли это, что ЭВМ считает процесс осушения целесообразным? Относительно рек — да. Но может быть пропадут какие-то уникальные виды флоры и фауны? Об этом машину «забыли» спросить. А ущерб может быть непоправимым.

Значит, чтобы ЭВМ дала ответ, мы должны сами заранее знать все, что существенно для выбора правильного решения. Разве это возможно? В полной мере — нет. Наши знания никогда не бывают исчерпывающими. Однако и машина ничего не «знает» кроме того, что в нее вложил человек. Она лишь воспроизводит заложенное в ней решение на основе конкретных данных. ЭВМ, располагая большим количеством описаний и различными процедурами компиляции, может получать и новые тексты, которые не были целенаправленно в нее введены. В частности, какие-то из них могут содержать принципиально новые знания или шедевры изящной словесности. Однако сама ЭВМ не в состоянии отличить такой шедевр от бессмысленного текста, поскольку значимость результата

работы ЭВМ определяется практикой в реальном мире — активной деятельностью человека в природе и обществе. Если бы в ЭВМ была заложена полная модель природы и общества с точки зрения интересов каждого человека, тогда ЭВМ могла бы иметь критерий для критического восприятия своих новых текстов. Эта полнота, разумеется, не достижима, но неуклонное приближение к ней, по-видимому, и есть главная цель информатики. Именно об этом нужно помнить и не надеяться, что наступит время, когда все возьмет на себя машина. Впрочем, такая жизнь была бы неинтересной. Поэтому не будем подобно страусу прятать голову в песок при столкновении с трудностями. Не бояться трудностей, смело идти им навстречу — вот, что возвышает человека. Труд, связанный с творческим познанием мира и ответственностью за жизнь и красоту Земли, никогда не будет отдан машине. Это тот труд, который позволит человеку всегда оставаться Человеком.

# ОГЛАВЛЕНИЕ

Предисловие . . . . .	3
Введение . . . . .	6
 1. ЭВМ ВЧЕРА И СЕГОДНЯ . . . . .	 10
1.1. Открытие и изобретение . . . . .	—
1.2. Мечты и реальность . . . . .	11
1.3. На пути к новой технологии . . . . .	14
1.4. Машина математиков . . . . .	18
1.5. Первые вычислительные машины . . . . .	19
1.6. ЭВМ сегодня . . . . .	22
 2. ЧТО ТАКОЕ ЭВМ? . . . . .	 29
2.1. Память . . . . .	—
2.2. Арифметико-логическое устройство . . . . .	38
2.3. Управление . . . . .	43
2.4. Ввод — вывод . . . . .	46
2.5. Вычислять, помнить и управлять . . . . .	51
2.6. Микропроцессоры . . . . .	53
 3. ЯЗЫК, ЧЕЛОВЕК, ЭВМ . . . . .	 57
3.1. Путь задачи к ЭВМ . . . . .	—
3.2. Язык и информация . . . . .	59
3.3. Язык, понятный ЭВМ . . . . .	61
3.4. Язык естественный и искусственный . . . . .	66
3.5. Постановка задачи для ЭВМ . . . . .	69
3.6. Трансляция и исполнение . . . . .	74
3.7. Какой язык лучше? . . . . .	78

4. ОТ ПОСТАНОВКИ ЗАДАЧИ К РЕШЕНИЮ ЕЕ НА ЭВМ . . . . .	80
4.1. Зачем нужны модели? . . . . .	—
4.2. Модель и алгоритм . . . . .	84
4.3. Приемы построения алгоритмов . . . . .	88
4.4. Искусство программирования . . . . .	89
 5. ПРОГРАММНЫЕ СРЕДСТВА . . . . .	 93
5.1. Элементы программирования . . . . .	—
5.2. Процедуры и функции . . . . .	94
5.3. Переменные и константы . . . . .	97
5.4. Циклы и условия . . . . .	98
5.5. Структуры данных . . . . .	101
5.6. Языки высокого уровня . . . . .	102
5.7. Программное окружение . . . . .	107
5.8. Средства общения с ЭВМ . . . . .	110
 6. ИНТЕЛЛЕКТ ЕСТЕСТВЕННЫЙ И ИСКУС- СТВЕННЫЙ . . . . .	 113
6.1. Может ли машина мыслить? . . . . .	—
6.2. Можно ли обойтись без человека? . . . . .	117
6.3. Чего не может ЭВМ? . . . . .	120

НАУЧНО-ПОПУЛЯРНОЕ ИЗДАНИЕ

Виктор Васильевич Александров,  
Виталий Николаевич Арсентьев,  
Анна Владимировна Арсентьева

## ЧТО МОЖЕТ ЭВМ?

Редактор *В. И. Важенко*  
Художественный редактор *С. С. Венедиктов*  
Технический редактор *П. В. Шиканова*  
Корректор *Н. Б. Старостина*  
Обложка и рисунки художника *Г. Г. Светозарова*

ИБ № 5101

Сдано в набор 26.03.87. Подписано в печать 29.09.87. М-18500.  
Формат 84×108<sup>1</sup>/<sub>32</sub>. Бумага кн.-журн.  
Гарнитура литературная. Печать офсетная. Усл. печ. л. 6,72.  
Усл. кр.-отт. 27,2. Уч.-изд. л. 6,89. Тираж 200 000 экз.  
Заказ 935. Цена 40 коп.

Ленинградское отделение ордена Трудового Красного Знамени издательства «Машиностроение». 191065, Ленинград, ул. Дзержинского, 10.

Ордена Октябрьской Революции, ордена Трудового Красного Знамени Ленинградское производственно-техническое объединение «Печатный Двор» имени А. М. Горького Союзполиграфпрома при Государственном комитете СССР по делам издательств, полиграфии и книжной торговли. 197136, Ленинград, П-136, Чкаловский пр., 15.

## УВАЖАЕМЫЙ ЧИТАТЕЛЬ

*ЛЕНИНГРАДСКОЕ ОТДЕЛЕНИЕ ИЗДАТЕЛЬСТВА «МАШИНОСТРОЕНИЕ» ПЛАНИРУЕТ ВЫПУСТИТЬ СЛЕДУЮЩИЕ КНИГИ ИЗ СЕРИИ «НАУЧНО-ПОПУЛЯРНАЯ БИБЛИОТЕКА ШКОЛЬНИКА»:*

**Александров В. В., Шнейдеров В. С.** Рисунок, чертеж, картина на ЭВМ/Под общ. ред. В. В. Александрова.— Л.: Машиностроение, 1987.— 7,5 л.: ил.

Рассказано об электронно-вычислительных машинах, которые умеют выполнять чертежи, рисовать и генерировать цветные изображения. Кратко описано их устройство. Показано, как можно просто дать задание ЭВМ и использовать ее в качестве инструмента конструктора, проектировщика, инженера и специалистов других профилей. Приведены примеры использования «электронных художников» на производстве, в архитектуре, искусстве. В приложении представлены репродукции графических работ, выполненных ЭВМ как по заказу человека, так и самостоятельно.

Изложение построено на базе материала программы средней школы.

**Александров В. В., Арсентьев В. Н.** ЭВМ: игра и творчество.— Л.: Машиностроение, 1988.— 8 л.: ил.

Рассмотрено использование персональных компьютеров для игр, способствующих стимулированию творческого воображения, т. е. интеллектуальному тренингу. Описано применение компьютерной техники для имитации космических полетов, съемки научно-фантастических фильмов, создания новых звуковых форм.

Приведены примеры организации популярных игр на ЭВМ и принципы алгоритмического построения с помощью ЭВМ игровых и творческих компонентов.

## **Уважаемый читатель!**

**Издательство просит Вас в прилагаемой анкете отметить позиции, соответствующие Вашей оценке этой книги.**

**1. Появилось ли у Вас желание иметь личный персональный компьютер?**

**Да**

**Нет**

**2. Какой процент новой информации почерпнули Вы из книги?**

**100 %**

**75 %**

**50 %**

**25 %**

**0**

**3. Ваша оценка содержания книги.**

**Отлично**

**Хорошо**

**Удовлетворительно**

**4. Помогла ли Вам книга в повседневной работе или учебе?**

**Да**

**Нет**

**5. Какие по Вашему мнению нужны научно-популярные книги по компьютерной технике?**

---

---

---



Фамилия . . . . .

Имя . . . . .

Отчество . . . . .

Возраст . . . . .

Образование . . . . .

Кем работаете (где учитесь)? . . . . .

. . . . .

Кем хотели бы работать? . . . . .

. . . . .

Просим отрезать страницу по линии отреза и в почтовом конверте выслать по адресу:  
**191065, Ленинград, ул. Дзержинского, 10, ЛО изд-ва «Машиностроение».**

*В. В. Александров, В. Н. Арсентьев, А. В. Арсентьева.*  
**Что может ЭВМ?**

# Что может ЭВМ?

**ЭВМ играет, ЭВМ проектирует, ЭВМ рисует, ЭВМ ставит диагноз, ЭВМ..., ЭВМ..., ЭВМ...**

**ЭВМ как мощнейший инструмент для усиления интеллектуальной деятельности человека, изменяющий всю инфраструктуру информационного взаимодействия в обществе, подобно урагану врывается в нашу повседневную жизнь. Персональный компьютер стал более престижным, чем автомобиль. В чем же причина этого феномена? Как устранить психологический барьер, обусловленный, с одной стороны, нежеланием объективно разобраться в сильных и слабых сторонах ЭВМ, а с другой — фетишизацией ее возможностей, роли и места в жизни общества!**

**Эти проблемы, стоящие сегодня на повестке дня, а также огромный, все возрастающий поток рекламных и популяризационных выступлений по радио и телевидению, публикаций в периодической печати в духе научной фантастики побудили авторов на основании собственного опыта работы с ЭВМ изложить свой взгляд на то, что же такое ЭВМ и что она может.**

